;login:

The USENIX Association Newsletter

Volume 7 Number 4 September 1982

CONTENTS

The deadline for submissions for the November issue of *;login:* is October 29, and for the December issue is November 30.

NOTICE

;login: is the official newsletter of the USENIX Association, and is sent free of charge to individual and institutional members of the Association.

The USENIX Association is an organization of AT&T licensees and sub-licensees formed for the purpose of exchanging information and ideas about the UNIX* operating system and the C programming language. It is a not-for-profit corporation incorporated under the laws of the State of Delaware. The officers of the Association are:

Membership information can be obtained from the Association Office:

USENIX Association
P.O. Box 7
El Cerrito, CA 94530

The Office phone number was not available at publication time; it should be available from information (415 area code) by late October.

Members of the UNIX community are heartily encouraged to contribute articles and suggestions for *;login:*. Your contributions may be sent to the editor electronically:

ucbvax!g:usenix

or through the US mail to the Association Office at the address above.

The USENIX Association reserves the right to edit submitted material.

;login: is produced on UNIX using *troff* and a variation of the −me macros. We appreciate receiving your contributions in *n/troff* input format, using any macro package. If you contribute hardcopy articles please leave left and right margins of 1" and a top margin of 1½" and a bottom margin of 1¼". Hardcopy output from a line printer or most dot-matrix printers is not reproducible.

---

*UNIX is a trademark of Bell Laboratories

UNICOM — The Next Meeting

The next general meeting of the USENIX Association will be a joint meeting with /usr/group and the Software Tools Users Group. It will be called **UNICOM** and will be held in San Diego, California, Tuesday, January 25 through Friday, January 28, 1983, at the Town and Country Hotel. The meeting is being hosted by the University of California at San Diego, with Tom Uter acting as the overall local conference coordinator.

The local arrangements chairperson is

Judy DesHarnais
UNICOM
P.O. Box 385
Sunset Beach, CA 90742
(213) 592-3243

The technical program chairpersons for each group are:

Bill Appelbe of UCSD for USENIX
Joel Carter of the Wollongong Group for /usr/group
Dave Martin of Hughes Aircraft for Software Tools Users Group

Local arrangements are being handled by

Harry Kerman
Conventions West Inc.
9301 Wilshire Blvd. Suite 203
Beverly Hills, CA 90210
(213) 278-2326

A meeting pre-announcement is being mailed to an extensive list the week of September 27. (Thank you to Pacific Software Manufacturing Company of Berkeley for use of a printer to print about 3000 labels from the USENIX data bases.) If you do not receive the meeting pre-announcement and need registration and/or accommodation information contact the local arrangements chairperson.

For information on reserving a booth in the vendor display contact Conventions West.

System III Educational and Administrative Licenses Now Available

AT&T has announced the release of System III for domestic *and* foreign educational institutions. The cost will be $800 for the first CPU. For each additional CPU the license will cost $400 and there will be an additional charge of $400 if the tape and documentation set is provided by Bell. (There are five distribution tapes, each tailored to different models of DEC CPUs.)

Existing educational license holders may upgrade all licensed CPUs to System III for a one-time charge of $800. One tape and documentation set will be provided. If additional distributions are needed there will be a $400 charge for each.

Administrative licenses for System III will cost $16000 for the first CPU and $5400 for each additional CPU. Existing administrative license holders will get full credit for their current license.

These licenses are handled by Nina McCloskey and Bob Hoffman at

AT&T
Technology Licensing
P.O. BOX 25000
Greensboro, NC 27420
(919) 697-5081

The New USENIX Association Office

The USENIX Association solicited proposals for the provision of office services to the Association in the last issue of *;login:* and at the Boston Meeting. Five proposals were received. The Board of Directors, at its meeting of August 18 and 19[†], selected the proposal presented by Penny Penny and Strong (**PP&S**) to establish and maintain the USENIX Association Office.

The Office will provide membership services and Board support to the Association. These services will include editing and distribution of *;login:*, editing and distribution of the Association software distribution tapes, reproduction and selling of UNIX-related documents, maintaining membership records, and such other services as the Board may request.

The Board has placed the responsibility for the proper functioning of the Office in the newly-defined position of Association Executive Director. The Executive Director will not be a member of the Board, although this person will attend Board meetings. The new Executive Director is Tom Strong. Mr. Strong will also continue to be the Editor of *;login:*.

The contract was actually awarded in late-September. PP&S will have the Office open and available for business on October 25, 1982. The Office manager is Alice Penny. The Office may be contacted by US mail at

USENIX Association Office
P.O. Box 7
El Cerrito, CA 94530

The Office will be available by phone weekdays a minimum of four hours a day, 10am to 2pm Pacific Time. A phone-message machine will be available when there is no one in the office. The new phone number was not available at publication time; it should be available from information (415 area code) by late October. The Office may also be contacted electronically at

ucbvax!g:usenix

PP&S is strongly committed to providing timely and useful services to the Association. We urge you to communicate your needs and your contributions for *;login:* and the software distribution tapes to the Office.

. . . Lou Katz
. . . Tom Strong

[†]summarized in another article in this issue of *;login:*

Reports on the USENIX Board of Directors Meetings

Open Meeting of the Members with the Board at Boston

Boston, Massachusetts

The Board held an open meeting on July 8, 1982, during the Boston conference to receive input from Association members. The major points of discussion are listed below.

- Concern was expressed about the dilution of the "technical" sessions, and about technical presentations that turn out to be product promotions.
- There was often difficulty seeing the slides and especially the viewgraphs; multiple screens [and more readable graphics...Ed] were recommended for presentations to the whole group.
- The availability of the abstracts at registration was felt to be very useful. The Board felt that refereeing of papers before the meeting was not possible because of the "timeliness" of the material presented.
- There was discussion of the length of the conference and the notion of parallel sessions. The members present favored a conference length of three days over five days by 2 to 1. The idea of parallel sessions to allow enough time for all the talks offered met very mixed and strong feelings.
- Concern was expressed over the lack of guidelines for the presentations and papers.
- It was suggested that the session chairpersons receive an honorarium and that they produce synopses of the talks in their session.

. . . John Donnelly

July Board Meeting

Boston, Massachusetts

Two Board meetings of the USENIX Association were held during the week of the Boston Conference, the first on Monday, July 5th to deal with items concerning the meeting and any business concerning transfer of operations to the newly elected Board, and the second on Friday, July 9th and Saturday, the 10th. During the Boston Conference there was also a meeting of the Board with members[†].

Monday, July 5th, 1982

The meeting was brought to order at 7:10 pm. Board members present were: Katz, Donnelly, Ferentz, Law, Scherrer, and Wedel. There were seven invited guests: Suzanne MacNary, Local Arrangements Chairperson of the Boston meeting; Board members-elect Bruce Borden, responsible for the seminar program, and Alan Nemeth, Chairperson of the Technical Program Committee; members of /usr/group Bob Marsh, Bob Greenberg and Lizabeth Riley; and Neil Groundwater of Software Tools.

The forthcoming meeting was a joint meeting of the three groups; the USENIX Association, /usr/group and Software Tools. The arrangements were complicated by Monday July 5th being a holiday, constraining scheduling of sessions and setting up of vendor booths.

[†]reported in the previous article

a. The Boston Meeting

There were no major problems to be dealt with. At this meeting for the first time, a seminar would be held in which material covered by AT&T licenses would be discussed (4.2BSD). There were minor problems with checking of eligibility, but Borden felt these could be handled without too much problem.

Expenditures for the conference appeared to be slightly below the budgeted amount, despite unanticipated costs for production of the abstracts and somewhat higher than expected costs for the reception to be held at the Aquarium. There would be 41 vendors occupying 55 booths, and 825 people had preregistered.

The room assignments for Friday were questioned and changed — the Software Tools meeting was best suited to the 400 seat room at the Sheraton. This was seen as a problem by Software Tools, but there did not seem to be an alternative.

There appeared to be several problems in planning for a joint meeting. Nemeth felt that there was a real need for more feedback on parallel sessions and their contents. Greenberg asked if it was appropriate to comment on why the present conference was a "failure as a joint enterprise", giving the following reasons: the meeting was never publicized as a joint meeting; /usr/group had its own way "of doing things wrong"; the 20 minute duration set for most papers would not be appropriate for some of the normal /usr/group talks; and that the three /usr/group presentations were scheduled for late Friday afternoon.

Nemeth felt that this was due to a difference in style of getting papers submitted for presentations — there was a sufficiency to fill the available time submitted from USENIX members resulting just from the call for papers. /usr/group normally asked for papers by telephone contact with possible authors. Katz had seen the preliminary schedule and had discussed the lack of /usr/group submissions with Nemeth, who passed this on to Marlene Martin. It did not seem to be a problem, and the schedule was left to Nemeth.

Nemeth thought there was another problem, that of lack of variety in the sessions — there were no panel discussions for instance. In addition, he felt that the members of the program committee had to have far more interaction, with strong input from each group involved, and that this could only easily be achieved if the members lived in reasonable proximity, although there was no need for the committee to be local to the conference site.

Marsh summarized his feelings by saying that the real problems for future joint meetings were the "nuts and bolts", (splitting of revenues, possible alternation of organization of joint meetings and programs) and that he felt these would be worked out. There was general agreement that more interaction in the planning stages between the three groups would have been useful.

Katz felt there was a real problem in the way housing was handled, in that no conformation of room bookings were sent out, due to the hotels and Rogal, Inc. each assuming that this was being done by the other. Fortunately, there were plenty of rooms available, so it was not a disaster.

Katz, as President of USENIX, proposed a vote of thanks to the retiring Board members, Mel Ferentz, Ira Fuchs and Peter Weiner. A recess was called with the meeting to reconvene at 9 pm in executive session.

b. Minutes of the previous meeting

Scherrer asked that the minutes be amended to show that Bob Marsh had agreed to produce the proceedings of the Boston meeting. The amended minutes were accepted.

c. Report by the Treasurer

Ferentz presented a financial summary and an audit of the books by Breiner and Bodian, certified public accounts. As he was retiring as Treasurer of the organization, methods of orderly transfer of duties to the new treasurer (Tom Ferrin) were discussed and decided upon. The transfer was complicated by Ferrin's absence from the meeting due to prior commitments in Europe, and the setting up of a formally organized USENIX Office, for which proposals were being requested.

There were two questions on the financial audit: an item was listed as "Sales of Software", which should have been listed as income from distribution of previous years software tapes; the item listed as "Capital Stock" represented funds brought into the USENIX Association when it was founded from the preceding UNIX Users group.

d. Any other business

It was proposed that, in view of Mel Ferentz's contributions to the USENIX Association and its predecessors, he be granted Life Membership in the Association. The motion was passed unanimously.

Scherrer stated that monies due to Software Tools from the Santa Monica and Austin meetings had not been received. Arrangements were made for the money to be transferred.

The meeting was adjourned at 11 pm.

Friday, July 9th, 1982

The meeting of the newly elected Board was brought to order at 6:30 pm, Friday, July 9th, 1982. Board members present were Katz, Borden, Donnelly, Law, Nemeth, Scherrer, and Wedel. There were six invited guests: Suzanne MacNary, Local Arrangements Chairperson of the Boston meeting, members of /usr/group Bob Marsh, Bob Greenburg, and Lizabeth Riley; Randy Frank (?) and Neil Groundwater of Software Tools.

e. Meeting Postmortem

Final registration figures for the conference were 1230 attendees, of which 825 preregistered. 315 rooms were taken at the Copley and a number that varied between 325 and 375 at the Sheraton. The following were thought to have been problems and needed attention for future meetings:

too many booths in the available vendor exhibition space;
lack of confirmation of room bookings;
lack of public seating space in the hotel lobby;
lack of lower priced rooms or information as to their possible availability;
inadequate size of the message board and poor siting of the jobs available board;
poor publication of changes in schedules and session locations;
arrangements and advertising of BOF sessions was not good;
a need to instruct session chairpersons in the use of the session timer, etc.;
a distinct lack of danish during the coffee breaks.

The general feeling was that the Audio/Visual operator had given excellent service, but that the display screen should have been higher to improve visibility, that more microphones were needed and that better enforcement by the session chairpersons of use of microphones during the question periods was necessary.

Nemeth suggested that the format for the next conference, if held jointly, be Software Tools sessions in parallel with tutorials preceding the meeting, with three days of USENIX sessions in parallel with /usr/group sessions. There was general agreement on this format.

Katz summarized the meeting of the USENIX Board with the membership. Strong feelings were expressed at the meeting concerning the need for differentiation between technical and commercial sessions. In fact, none of the sessions were characterized as to content — it would be useful, as in the past to divide them into graphics, communications, vendor issues etc. A straw vote showed there was strong feeling in favour of three day meetings with parallel sessions. In spite of the amount of griping about "commercial sessions", a further straw vote elicited the surprising result that there was a 2 to 1 preponderance in favour of outright merger with /usr/group.

f. Future cooperation with other groups

A motion was proposed and approved that USENIX pursue a joint meeting with /usr/group and Software Tools in San Diego in January, 1983.

It was thought that there would be little further feedback from /usr/group as to whether it wished to participate in further joint meetings until the organization held its first Board meeting after the recent elections. The meeting was scheduled for July 21th.

At this point there was a short break for dinner, after which all guests withdrew.

g. Budgets

In view of the unavoidable absence of the newly elected Treasurer, Tom Ferrin, it was agreed that Katz, Ferrin and Law should meet in Boston during the SIGGRAPH meeting to review the situation.

h. Temporary Services for USENIX

Until a formal office has been set up, it was agreed that Law would handle only necessary correspondence, but not attempt to provide full office services. Requests would be acknowledged, but in all probability would not be serviced until the office is functional, hopefully some time in September. Scherrer would continue to act as the Chairperson of the Tape Distribution Committee, and would complete, if possible, the distribution presently in progress.

i. The Next Board Meeting

The next Board meeting would normally deal with the San Diego meeting arrangements. However, the most urgent item on the agenda will be a review of the proposals submitted for the USENIX Office, followed by selection (and final negotiation if necessary) and implementation of the service. It was agreed that a subcommittee of Donnelly, Borden and Scherrer be appointed to deal with the San Diego conference arrangements, and that the Board would meet August 18th and 19th in San Francisco to deal with the more pressing matters.

The meeting adjourned at 11 pm, to reconvene at 9 am the following day.

Saturday, July 10th, 1982

j. Specifications for Office Services

Law and Wedel had each produced and circulated a skeleton of requirements for office services, and Scherrer had defined procedures for handling distribution tapes. Many additional items were added, and a consensus emerged that the function provided by an executive director or executive secretary was necessary. It was agreed that a Board member should have responsibility for the office, with the Vice-President being assigned the task.

k. New Committees

The following committees were appointed:

(1) Office Selection Committee: Katz, Nemeth and Wedel. The charge was to write a request for proposals, mail these to possible bidders and anyone requesting copies, evaluate the responses and make a recommendation to the next Board meeting. The RFP should be mailed by July 19th, bids to be received on or before August 9th, and the analysis presented to the Board August 18th.

(2) Newsletter Committee: Borden, Katz, and Nemeth.

(3) Distribution Tape Committee: Scherrer plus co-opted members.

(4) Program Committee for San Diego Conference:

(5) Other Organizations: Borden, Donnelly and Scherrer to deal with matters concerning other UNIX groups, local interest or users groups and UNIX meetings in other countries.

It was agreed that an Audit Committee would be discussed when Ferrin was available. A Bylaws Committee would be set up at the next Board meeting, and that a timetable should be established for a complete review and revision.

l. Newsletter Status Report

As the production and distribution of the Newsletter had been incorporated in to the Office Services Specification, it was decided to continue with present production methods if possible until the end of this calendar year.

m. Future USENIX Meetings

Arrangements are already well under way for the San Diego meeting in January. It was felt that if meetings were to be held jointly with other groups, they must have input into decisions. Rogal, Inc. (the conference support company hired for the Boston meeting) had been asked to submit a proposal for providing services at the Toronto meeting to be held in the summer of 1983.

n. Any Other Business

A revision was made in the proposed distribution between the three participating groups at the Boston conference of any surplus revenue, and a motion made and passed that the distribution should be as follows: 50% to USENIX, 30% to /usr/group, and 20% to Software Tools.

The meeting adjourned by general agreement at 12 noon.

. . . Lew Law
Secretary of USENIX

August Board Meeting

Burlingame, California

Wednesday, August 18, 1982

The meeting was brought to order at 2:09 pm, Wednesday, August 18th, 1982. The agenda for the meeting was set: 1) meeting items, 2) financial, and 3) office. All Board members except Law were present. Borden taking minutes.

Donnelly, Borden and Scherrer summarized the meeting with /usr/group held the same morning, August 18th. There seemed to be no snags dealing with /usr/group, although there were some outstanding issues still to be resolved. The name of "UNICOM" was discussed as the title for the upcoming San Diego conference.

The discussion turned to the Boston conference and future conference items. Wedel requested more scribe and proceeding coverage in the future. Approximately 30 papers were in the hands of /usr/group for preparation of Boston conference proceedings. Proposed schedule for UNICOM:

Tuesday:	Education/Tutorials/Software Tools
Wednesday:	AM: Joint, PM: Parallel
Thursday:	Parallel all day
Friday:	AM: Parallel, PM: Joint closing

It was also decided to suggest to /usr/group that the vendors set up Monday and be open all day Tuesday specifically for the conference attendees — perhaps, even close the vendor booths to the public for Tuesday.

A 42.5%—42.5%—15% revenue sharing for USENIX, /usr/group, and Software Tools for the San Diego meeting was accepted.

The discussion then focused on the desired and expected size of UNICOM. /usr/group would prefer 5000 attendees, whereas, USENIX would prefer 1200-1500, keeping the size of sessions more reasonable.

Each speaker will get a bright "speaker's" ribbon in their registration packet. Legitimate press personnel will be given a "press" ribbon.

The name UNICOM was formally accepted as the name for the San Diego conference.

Ferrin will be responsible for procedures for a UNICOM bank account, with input from USENIX, /usr/group and Software Tools. Katz suggested audio visual support from the group which handled SIGGRAPH for approximately $10K, which other members rejected.

Borden was authorized to purchase a terminal, printer and modem for use by UNICOM.

Borden's suggestion to purchase a small UNIX computer from a vendor offering a good discount was tabled until the office was settled.

Summary of the Boston meeting was discussed. More details on income and expenses were necessary; Ferrin is to obtain them from Rogal and MacNary. A more detailed accounting with different categories would be requested. Even though not complete, the current accounting would be given to /usr/group.

Rogal visited Toronto for reconnaissance of the summer 1983 meeting. They prepared a proposal for managing the Toronto conference. Rogal and Conventions West were discussed and compared. /usr/group is to be given a copy of the Rogal proposal. Escape clauses are needed in all future contracts.

Possible sites for future meetings: D.C., Chicago, Portland (Tektronix host), and others. Rogal and Conventions West should propose sites which are appropriate to the size and style of our meetings.

The discussion turned back to San Diego topics. A contract with Judy DesHarnais to act as local arrangements chairperson for UNICOM was approved.

Nemeth summarized the office proposals. Since Donnelly submitted a proposal, this summary was very general. Donnelly will not attend the meeting Thursday morning while the proposals are discussed.

Borden will have a summary of this and the joint USENIX, /usr/group and Software Tools committee meetings ready by the end of the month for *;login*.

Wedel and Law are a committee on bylaws — to have background and some suggestions ready for next board meeting. Next board meeting to be held November 4th and 5th, 1982, in the Bay Area to minimize travel.

Closing discussion: member and student discounts will be offered at UNICOM and there will be a high late registration fee. The joint committee will discuss access to the conference mailing list with /usr/group. More information (such as various network addresses) should be included on conference attendee list.

Thursday, August 19, 1982.

Meeting reconvened at 9:00 am.

The morning was devoted to discussion and review of the 5 proposals received for office services. Donnelly was not present as he had submitted one of the proposals. By noon, two of the five proposals had been eliminated, and a straw vote on the remaining 3 proposals divided two votes each.

It was suggested that USENIX accept the proposal of Penny Penny and Strong, as it was either first or second choice for all board members. This was approved and the office committee was directed to meet with PP&S regarding various questions and possible changes in their proposal. A meeting was set up for the following morning, Friday, August 20, 1982 at the Hyatt Burlingame.

The office negotiating committee was authorized to contract for legal services in connection with securing a contract for office services.

The president and treasurer were authorized to jointly sign a contract for office services on behalf of the USENIX Association according to the terms agreed upon by the negotiating committee.

The USENIX office should look into 800 phone service, as well as alternate discount telephone services such as Sprint and MCI.

The discussion returned to UNICOM. Various guidelines, requirements and preferences were conveyed to the conference committee. The board felt very strongly that there should be one admission fee for all members of any of the three groups, with non-members paying a higher fee. There should be student discounts, how much and whether only for pre-registration negotiable with /usr/group. Borden was given a mandate to negotiate this with /usr/group and Software Tools.

Agenda for next meeting:

- Next years dues
- Bylaws changes (Wedel)
- Meeting (conferences) report (Donnelly)
- Office report (Katz)
- Budget for 1983 (Ferrin)
- Manuals (Katz)
- Distribution Tapes (Scherrer)
- Treasurer's Report (Ferrin)
- Minutes
- Plan next meeting
- Future conference sites
- Cooperation with other groups
- Status of /usr/group relationship (Borden)

Scherrer reported on the distribution tapes. Approximately 200 1982 tapes have been sent. There were 5 new submissions from the Boston meeting.

More discussion of future sites: D.C., Atlanta, Chicago, Orlando, Phoenix, Portland, Denver, Anaheim, Seattle. D.C. and Portland are of most interest.

A suggestion was made to push the summer conference earlier, to avoid the summer heat and to move further away from SIGGRAPH. Borden to talk with /usr/group regarding future sites and schedules.

Ferrin to decide how to quickly and cleanly close out the USENIX N.Y. bank accounts and set up new accounts in the Bay Area.

. . . Bruce Borden

Notes on the Boston USENIX and /usr/group Joint Meeting July 1982

Notes on the Boston USENIX and /usr/group Joint Meeting

Editor's Note

Tom Strong

The following articles were written to summarize the main points of the presentations made at the Boston meeting for people who did not attend the meeting. Copies of the abstracts, viewgraphs foils, slides, and/or papers were used when available. Suggestions and corrections should be sent to me at the USENIX Association Office or electronically at

ucbvax!g:usenix

Tuesday, July 6, 1982

Session Chair: John Donnelly, NCAR

Embedding UNIX in a Product (or, is "Real-Time" Real?)

William R. Northlich Jr. and *T.D. McCreery*
Zehntel, Inc., 2625 Shadelands Dr., Walnut Creek, CA 94595
P.M. Powers
Megatest Inc.

This talk described Zehntel's experiences using UNIX as the operating system for its circuit board test systems. The software in Zehntel's systems serves two functions: it provides the development environment for programmers to write programs for each type of circuit board to be tested and it commands the hardware for the actual board testing. When previous software had been pushed beyond its limits the decision was made to try UNIX. They felt that UNIX's portability, development capabilities, and basic simplicity of structure would make it viable for their needs, in spite of its reputation as not being suitable for real-time applications.

The notion of "real-time" was discussed at some length. It was pointed out that the term has no rigorous definition; it connotes speed and is normally tied to some notion of an on-going (external) physical process. The notion that a real-time system is one that has the design goal of

Capacity > Load

seemed appropriate. That is, the system is designed to keep up with interrupts, even under the heaviest loads.

Zehntel found that they had to split their main tester program into two communicating processes due to address space constraints. This caused a significant slowing of the system test speed due to communications overhead. They first tried to overcome this by using pipes, then went to using shared memory by reserving two buffers from the kernel buffer pool and using a *phys* call to map them to a particular address inside each process. They also implemented system calls that allowed each process to call the other with a minimum of overhead. These changes caused the tester to run almost two orders of magnitude faster. The changes were relatively easy to make and could be made without changing the basic algorithms of the system because they were using UNIX both as the development system and in the target system.

The speaker has submitted a paper for the Conference Proceedings.

Implementing a Multiple-Process Real-Time System Under UNIX

A.V. Hays, Jr., B.J. Richmond, and *L.M. Optican*
National Eye Institute, National Institute of Mental Health, 9000 Rockville Pike, Bethesda, MD 20205

REX (Real-time EXperimentation) is a system for real-time laboratory data acquisition and control. It is used to sample up to eight analog channels and sixteen digital I/O channels, to control the subject through behavioral paradigms, to generate and present stimuli, and compute on-line graphics. The REX functions are divided among various cooperating processes: a "comm" process to control keyboard interaction, a "scribe" process to control files and write data to disk, an "int" process to respond to interrupts from clocks, analog-to-digital converters, etc., and a "display" process to generate the graphic displays. There is also a shared data area. Laboratory control is accomplished with a state-based interpreter in the "int" process.

REX runs on a dedicated machine. Three enhancements were required to UNIX in order to implement REX. These were for shared incore data segments, improved interprocess communication, and direct user response to device interrupts with minimum latency. The modifications they made were tailored to smaller PDP-11s* such as the 11/23, 11/34, and 11/40.

In order to provide response to device interrupts they lowered all kernel priorities from 7 to 6 except the device interrupt which is set at 7. This routine performs several functions including sampling and communication with other processes but it may not execute system calls.

The REX prototype was completed in March, 1981. The kernel is used at five sites running REX and one other site running a single-process real-time application.

A paper on the REX system is being prepared for WESCON.

Real-Time Performance

Johann George
Mark Williams Company, 1430 West Wrightwood Ave, Chicago, IL 60614

[I missed this talk and there was no paper submitted; the abstract is reproduced below...Ed.]

One of the most important uses of operating systems is in real-time applications. Several real-time operating systems exist. The problems arise when one desires a real-time operating system that is portable. The presentation will contain a discussion of the problems involved and the efforts made to solve them. Some of the problems discussed will be reducing interrupt lockout time and allowing user programs more control over input/output operations.

Real-Time Systems

Jim Isaak
Charles River Data Systems, Natick, MA 01760

Definition:

A Real-Time System acquires data and processes it at execution time.

This talk described the range and characteristics of real-time systems. In a real-time application the data is available only after the process begins. The process is event-driven, either by a clock or asynchronously by the data. Data rates vary enormously, as does the amount of computation required for a given amount of data. Examples at the extremes include radar and CAT scan processing, environmental monitoring, communications control, and things like data entry and word processing.

When planning a real-time system there are several things about your needs that must be considered:

(1) data collection rates;

*PDP is a trademark of Digital Equiptment Corporation.

(2) memory capacity needed;

(3) computational requirements;

(4) I/O requirements in type, speed, and volume; and

(5) what needs to be done at the same time.

These define a context for evaluating how any system will fill your need. If no system can do the job (with margins) then division into separate processors, custom hardware, and/or custom systems may be appropriate.

Mr. Isaak listed several things to look for when evaluating the hardware and software of a real-time system:

address space,
shared data space,
processor speed,
vectored interrupts,
bus levels and loading,
priority control,
multitasking and multiprocessing,
system overhead when changing processes,
user device drivers,
reliable synchronization techniques,
queuing mechanisms for messages,
exception handling,
file system speed and reliability, and
process residency locking.

The "UNOS" operating system offered by Charles River Data Systems has been designed for the type of real-time functionality described above. It is implemented on a 68000 with VERSAbus*.

Session Chair: Joseph Yao, Science Applications, Inc.

Optimizing Database Queries in SQL

Dennis F. Meyer
UNIQ Computer Corporation, 28 South Water St., Batavia, IL 60510

This talk described a UNIX-based system developed to optimize database queries written in the Structured Query Language (SQL). SQL was developed by IBM as the main interface to an experimental relational database system. The system handles a slightly modified subset of the query facilities of SQL. It has been ported to several different operating environments.

The compiler component of the system has three main phases: a parser, an optimizer, and a code generator. These phases communicate through intermediate human-readable ASCII files. The information passed consists of symbol tables, which contain information about the structure of the data, and operation records, which are used to form the abstract syntax tree. The parser uses a lexical analyzer generated by *lex* and a syntax analyzer generated by *yacc*. It diagnoses syntax errors, builds the initial abstract syntax tree, and builds the symbol table using a data dictionary that describes each relation and its attributes.

The optimizer makes improvements to the source queries by doing semantic analysis and making transformations to the abstract syntax tree. The transformations are designed to minimize the amount of work and number of decisions for the interpreter. Transformations include simplification of logical expressions, forcing data conversion when necessary, rearranging statements to improve the order of evaluation, and building a query dependency tree.

*VERSAbus is a trademark of Motorola.

The code generator translates the abstract syntax tree into binary object code for the interpreter. In order to maximize speed and efficiency the interpreter was implemented with four basic rules in mind:

1. don't do anything unless it is necessary;
2. once something must be done, do as much as possible with it;
3. use memory instead of disk, whenever possible; and
4. use known methods for writing efficient code.

Interfacing UNIX to Backend Database Machines

Michael E. Duffy
Consultant, 65 Franklin Street #24, Allston, MA 02134

This talk described an approach to handling large databases on UNIX that minimizes the changes to, and overhead in, UNIX. The approach is to use a separate dedicated database machine (a "back-end") to handle the database. The UNIX system(s) communicate their database queries to the backend using a well-defined protocol over a hardware interface. The backend passes back just the qualified data. This cuts down on both the I/O and CPU cycles of the "front end" machine.

Some of conditions where a backend database machine may be worth considering include:

— "lots" of data
— frequent access
— need for fast response
— multiple users of the system and of the database
— multiple hosts using the database
— need for reliability
— complex and/or *ad hoc* access to the database

In addition to reducing overhead in the front end a database backend can provide such facilities as concurrency control, audit trails, checkpointing, and crash recovery.

Mr. Duffy went on to describe connecting a Britton/Lee IDM-500 database processor and 5 PDP-11/70s running Interactive Systems' IS/1, a version of V6 UNIX. The IDM-500 can communicate with the front end(s) either through a bit-serial RS232 or byte-parallel IEEE-488 bus interface. The system being described uses National Instrument's GPIB11-2 UNIBUS-to-IEEE-488 interfaces on each 11/70. The interface and bus are capable of transmitting data at up to 500 Kbytes/second, although measured speeds have not yet approached that. The IDM-500 accepts queries in a parsed form of a language called IDL. Britton-Lee provides a *yacc*-based parser for this language, as well as a C preprocessor which allows IDL statements to be embedded in C programs. The software provided by Britton-Lee is designed to run under UNIX V7.

Several difficulties were encountered when converting the V7 device drivers to run on V6. Some of these are listed below for those who need to retrofit V7 drivers. [I may not have recorded them correctly...Ed]

- argument-passing conventions differ substantially (e.g., *u.u_r.r_val1* becomes *u.ar0[R0]*);
- unions are used in V7 so you have to figure out what the V6 name of the variable is (eg, *b_un.b_addr* becomes *b_addr*);
- V7 drivers use byte counts (e.g., *b_un.b_bcount*) and V6 uses word counts (e.g., *b_wcount*) but they are both *int* fields so just changing the name works fine;
- calls to *geteblk*() in V7 to allocate a buffer from the buffer pool become *getblk(NODEV)* in V6. If you ever intend to call *bfree*() on a block so allocated be absolutely sure that you put a **-256** back into *b_wcount* as V6 assumes that only disk devices ever allocate buffers.

The IDM has been quite successful. Mr. Duffy believes that it will support on the order of 10 transactions per second. There was one major oversight in the interface: the firmware as received did not allow multiple UNIX front ends! Britton-Lee is shipping new PROMs for the interface that will fix that problem.

Britton-Lee also offers the IDM-200, a smaller, less expandable model which provides the same functional interface as the IDM-500.

Time and Tuples: Concurrency Control in LOGIX

Fred M. Katz
Logical Software, Inc., 55 Wheeler St., Cambridge, MA 02138

This talk described "dated relations" where the time of insertion (and deletion, if applicable) of each item in a relation is maintained in the file with the relation. When all or part of a relation is locked for writing a new timestamp is assigned. When a relation is opened for reading it is possible to qualify the query with an "asof" phrase that specifies the time or date or database version. Thus it is possible to roll back to earlier versions of a database. Dated relations also help with concurrency control.

The amount of space that dated relations could use is potentially large. However, this must be compared to usefulness of having, and the space required for, previous versions of a database. If dated relations are used only for concurrency control and recovery it is possible to retain only the recent history of a relation.

Session Chair: Bill Joy, U.C. Berkeley

UNIX at Lucasfilm Ltd. or Does Darth Vader Code in C?

Bill Reeves
Lucasfilm Ltd., PO Box 2009, San Rafael, CA 94912

This talk began with a [neat!] display of graphics. UNIX is used for many projects at Lucasfilm. They run 4.1aBSD on VAX* 750s and 780s and use lots of disks. An extended file system using the networking code of 4.1aBSD has been developed. They have recently brought up V7 on a SUN 68000. Their implementation uses scattered memory allocation and has many features for their applications. The SUN does not require a local disk as they have implemented 3 and 10 Megabit Ethernet and have connected the SUN to their extended file system.

Other UNIX applications at Lucasfilm include: high-resolution three-dimensional computer synthesized color graphics, digital audio mixing and editing, video editing, database management, word processing, and computer video games development with Atari.

4.2BSD Overview

Bill Joy
U.C. Berkeley, Computer Systems Research Group, Computer Science Division, Berkeley, CA 94720

This talk presented an overview of work being done on UNIX 32V by the Computer Systems Research Group (CSRG) at U.C. Berkeley. The work is funded by an ARPA contract and will lead to an ARPA-standard VAX UNIX. The group distributes tapes with their modifications and additions (called Berkeley Software Distributions). Past distributions for the VAX have been 3BSD, which implemented virtual memory, 4BSD, which contained changes for fine tuning and job control, and 4.1BSD, which provided additional VAX support.

The current work is being done to support larger address space, high speed data access, and resource sharing on local networks. Changes are being made to some kernel facilities: a process will be able to be in several groups simultaneously; signals will have clean semantics and sophisticated stacks will be supported; there will be more precision in the timing routines and virtual time alarm clocks will be supported; and memory management will be changed to allow file pages to be mapped, pages to be shared, and page protection to be changed. The next full distribution, 4.2BSD, is scheduled for late 1982. Intermediate versions planned are:

*VAX is a trademark of Digital Equipment Corporation.

- 4.1a in April, with networking improvements;
- 4.1b in June, with a new file system;
- 4.1c in August, with improved inter-process communication; and
- 4.1d in October, with page-mapped virtual memory facilities.

4.1aBSD was running on a VAX at the conference.

Design papers are available from CSRG.

4.2BSD Interprocess Communications Primer

Bill Joy

Four reasons were presented for enhancing the UNIX communication facilities:

- to support communication between unrelated processes;
- to implement multi-process programs like distributed data bases;
- to provide access to communications networks;
- to implement local network services, such as "file servers", so it will be possible to share files between machines and run UNIX on diskless workstations.

The facilities being built are to be transparent (so processes on different machines can communicate) efficient, and compatible (so existing naive[†] processes can generalize to a distributed environment without change). The following table, reproduced from one of the viewgraphs presented, shows the approach being followed.

Need	Concept	Function
Handle networks with different set of protocols, addressing, conventions, etc.	communications domain	Implements standard semantics of communication and addressing, etc.
Descriptor representing endpoint of communication	socket	Abstracts object from which messages send and receive
Handle different semantics of communication	socket types	Allow different communication semantics to be handled, like different file types in file system
Locate endpoints of communication	socket address	Allows unrelated processes to locate each other
Be able to multiplex operations	*select* facility	Allows selection of action which will not cause blocking

Communications domains are used to model networks, network addresses, and protocols. The "UNIX" domain is used for communicating within a single UNIX system. Communicating in a another domain (for example, in a network the machine is connected to) begins by using a standard request to access that domain.

There are different types of sockets to implement different types of communication.

- **SOCK_DGRAM** sockets are use to send single unreliable messages called "datagrams". These sockets are never connected to another socket; rather they are used to send messages to a specified destination (sort of like *mail*). Mechanisms are provided to receive such messages.
- **SOCK_STREAM** sockets are used to send byte streams back and forth between two connected points (sockets). These are upward compatible with pipes. Mechanisms are provided to create a pair of connected sockets.
- **SOCK_SEQPACKET** sockets are used to exchange sequences of packets (rather than sequences of bytes) between connected sockets.

[†] "Naive" processes use standard input and/or output and do not do seeks or IOCTLs — processes like *cat*. "Sophisticated" processes manage other processes or use knowledge about specific devices — processes like shells and screen editors.

A UNIX process can be blocked by reading when there is no data or writing when no data can be accepted. A new *select* mechanism has been implemented to allow sophisticated processes to determine if a contemplated action will cause it to be blocked.

4.2BSD Network Communications

Sam Leffler
U.C. Berkeley, Computer Systems Research Group, Computer Science Division, Berkeley, CA 94720

[Unfortunately my notes on this talk were quite incomplete...Ed]

Network communications are being improved to support applications such as remote copy, remote login, network status, remote file system dump, and extended file systems. The goals of the 4.2BSD networking facilities are to provide:

- a uniform user interface to network communications;
- adequate performance for use in a distributed environment on a local area network; and
- an internal structure of that will allow support of multiple network protocols and varying network hardware.

A wide variety of network interface drivers already exist. The user interface is such that the user need specify only the final address of the action and does not need to worry about network-to-network interfaces.

Future work will be towards more protocol and hardware support, protocol performance comparisons, further evaluation of vendor hardware, and improving the performance of the extended file system.

4.2BSD File System

Kirk McKusick
U.C. Berkeley, Computer Systems Research Group, Computer Science Division, Berkeley, CA 94720

The 4.1BSD file system has been found to be too slow for some applications that use very large files. The reasons for the (relative) slowness, and the approaches taken are discussed below.

data blocks too small

Large blocks are very efficient for large files but tend to waste a lot of disk space when applied to a whole file system. They have implemented a file system that lets each file have one smaller block, with the rest much larger. Measurements of the total wasted space on one large VAX at Berkeley with file systems with different sized blocks were:

512 byte blocks	4.2%
1024 byte blocks	6.9%
2048 byte blocks	22.4%
4096 byte blocks	45.6%
4096/512 new file system	5.4%
8192/1k new file system	10.6%

data too disorganized on disk

Information about the machine, cpu, I/O channels, and disk has now been parameterized and is used in "policy" decisions to insure reasonable block layout on a disk. Related inodes are put close together (e.g., a directory and its files), and inodes are distributed throughout the disk. File data blocks are put in rotationally optimal positions.

The new file system provides very significant improvements in data throughput but further experience is needed. File system performance is now core-to-core copy limited.

In the process of this work improvements were made to codes such as *dump, restor, fsck,* etc. They also implemented long filenames (up to 255 characters) and increased the inode size from 2 to 4 bytes. Symbolic links across devices and machines were implemented using code from Dennis Ritchie.

4.2BSD Licensing

David Mosher
U.C. Berkeley, Computer Systems Research Group, Computer Science Division, Berkeley, CA 94720

CSRG makes its software available to sites with AT&T UNIX 32V or System III[†] licenses for a prepayable fee which is currently $600. A license agreement must be executed with the University of California. The distribution includes a complete set of Berkeley and Bell manuals and documentation on the Berkeley software. The distribution media can be either two 2400 foot 1600 bpi tapes or three RK07 packs, two of which must be RK07-EF.

As of July, 1982, there were 310 4.1BSD licensees running approximately 700 VAX systems. The licensees are about 80% domestic and 20% foreign, 70% universities and 30% commercial.

The distribution supports VAX 11/780s, 11/750s, and 11/730s with a wide variety of disk, tape, communication, and networking subsystems.

The 4.2BSD distribution is expected to be available this winter.

For further information write to

Jeri Kotani, Distribution Coordinator
Computer Systems Research Group
Computer Science Division, EECS
University of California
Berkeley, CA 94720

4.2BSD Questions and Answers

Joy, Leffler, McKusick and *Mosher*

[The following are notes on the responses to the questions asked about 4.2BSD; hopefully the questions can be inferred...Ed.]

- /usr/group is doing some documentation on the System III license and contract; it should be available soon.
- 4.xBSD license holders should not need a new license for 4.2BSD but there will be a $600 service charge.
- Upgrading to 4.2BSD will require dumping and restoring all files.
- There is a new *f77* — see the previous issue of *;login:*.
- Multiplexed files are gone.
- Code for the SUN 68000 will be *#ifdef*'d into 4.2BSD.
- It is about 50Kb bigger in memory.
- The file system does bad block handling.
- Expect to handle gigabyte processes in virtual memory.
- There will either be a new debugger or a new version of *f77* that fits with the existing debugger.
- They have been working on the kernel to make a base for developing applications; expect very few user-visible changes.
- There is some more file system robustness; especially with synchronous writing at critical points. Most of this was in 4.1BSD.
- Ingres is sponsored by a different organization; it will continue to be in the distribution until the Ingres group asks to have it removed.
- Some things will have to be recompiled.

[†]System III is a trademark of Bell Laboratories.

- *Ioctl* calls have been changed to the way suggested by Mark Horton.
- No named pipes: use the IPC stuff.
- C compiler unchanged (so far).
- There are no regular contacts between CSRG and the System V people at Bell.
- The extended file system currently assumes a global *uid* (i.e., on all machines); there will probably be a *uid*-mapper of some sort.
- There may be work going on elsewhere to make 4.2 run on the VAX 11/782, or other dual-processor VAXs.
- The user-contributed software distribution is floundering for lack of manpower. Perhaps some outside organization *like USENIX* will pick it up [emphasis mine...Ed].

Wednesday, July 7, 1982

Session Chair: Mike Zuhl, Tektronix

Ped - A Portable Editor

Mario Ruggiero
University of Toronto Computing Services, CSS, 10 Kings College Road, Room 4306, Toronto, Ontario, Canada M5S 1A1

[I missed most of this talk so the notes below are taken from the abstract, paper and slides presented...Ed.]

The VIVA Project at the University of Toronto has been developing a student work station built from "state of the art" components. A VIVA configuration consists of a local processor, CRT, disk storage, communications with a remote host for remote job entry, and a method for storing and transferring user files. Implementations have evolved from a LSI 11/2 running RT-11 to a LSI 11/23 running UNIX to VAXs running VMS.

One of the components of VIVA is Ped, a full screen editor. Ped was designed to provide a friendly, safe user environment, to be easy to learn, and to be portable to a wide variety of operating system environments. It provides a wide range of editing abilities and has *help* and extended keypad support.

Ped is expected to run on UNIX, VAX VMS, and the UCSD p-system. Four areas of portability were discussed.

CPU portability
: Ped is coded in a high level language — Pascal. They have tried to avoid using non-standard features of Pascal but have found the standard weak in some areas. Ped is expected to be able to run on small CPUs so it is kept smaller than 64Kb.

Operating system portability
: The operating system interfaces are implemented as a set of system dependent routines. The file handler implements character stacks and can access them one character at a time. The operating system is assumed to provide random-access files and to allow block I/O to those files. The terminal interface can read one character at a time and write one character or a string.

Terminal independence
: Ped is able to run with a variety of terminal types through the use of a subset of the *TermCap* mechanism. It uses a compiled database to optimize for speed and size. The compiler checks for correct syntax in the terminal description so Ped will not have to.

User community
: Users are also "portable". Ped is used by users with a wide range of experience for both text and program editing. It does not allow for customizing of the user interface by experienced users, although such provisions are being considered.

Ped has been implemented on UNIX V6 and PWB and on RT-11. In the UNIX implementation they had problems with raw mode I/O, the lack of look-ahead in the input stream, and with file ownership and protection.

In the future they plan to bring Ped up on VAX UNIX and possibly on VAX VMS and various micros. They have found that portability requires extra, often underestimated, efforts. Modularization has reduced the effort required to transport Ped to other environments.

The speaker has submitted a paper for the Conference Proceedings. It includes a table comparing some of the features available in Ped with their equivalents in Berkeley's *vi*.

Portability of C Language Programs

Chaim E. Schaap
Delft Consulting Corporation, 392 Bleecker St., New York, NY 10014

A truly portable program can run *unchanged* on systems with different hardware and/or software. This talk focused on the problem of trying to write portable C programs and noted many potential pitfalls a programmer needs to recognize. Problems related to portability to non-UNIX systems were included.

There is no standard C compiler or library that is truly portable[†]. The programmer must decide what (hopefully common) subset to use. The best source of information currently is Kernighan and Ritchie's book[‡]. It may be a mistake to infer that one's own C compiler and library is portable.

There are many differences in the hardware that can affect even standard C programs. Mr. Schaap listed the following potential pitfalls.

word/byte size and byte order
word/byte alignment requirements
signed/unsigned data
floating point format
arithmetic
overflow handling
memory addressing and size of address space
stack organization
machine character set
display devices
communications devices

There are also potential operating system differences.

kernel handling of I/O, processes, and users
system calls
command interpreter
C compiler
assembler/loader/link editor
libraries

And of course there are possibly legitimate differences between C compilers.

evaluation order of subexpressions
type conversions
logical/arithmetic operations
multicharacter constants
functions of a variable number of arguments[*]

To write portable C code one should consider the pitfalls mentioned above, avoid environmental dependencies by using *lint*, typing things carefully, and parameterizing things where necessary, and give careful consideration to the portability of I/O.

[†]Some specific difference between C compilers, and a useful new feature, were pointed out in the Languages Sessions at the Santa Monica meeting. Notes on those presentations were printed in *;login:*, Volume 7, Number 1.

[‡]Brian W. Kernighan and Dennis M. Ritchie, *The C Programming Language*, Prentice-Hall, Inc., 1978.

[*]He mentioned that there is a non-well-known header in System III that contains an almost-portable specification for writing functions with a variable number of arguments.

The Object Oriented Pre-Compiler: Programming Smalltalk 80 Methods in C Language

Dr. Brad J. Cox
ITT Programming Technology Center, 1000 Oronoque Lane, Stratford, CT 06497

[I missed this talk and there was no paper submitted; the abstract is reproduced below...Ed.]

This describes the Object Oriented Pre-Compiler, (OOPC, pronounced Oopsy), a pre-compiler and a library of routines for producing C language programs that operate according to the run-time conventions of the Smalltalk 80 language.

Smalltalk 80 and OOPC are two programming languages that offer Object Oriented Programming, a technique in which both data and the programs which operate on that data are designed, built and maintained as inseparable units called *objects*. This is a powerful technique for decomposing difficult software design problems into more manageable pieces, and is of great interest as a conceptual basis, or organizing principle behind the ITT Future Programming Environment.

Teaching *awk* as a First Programming Language

Bill Tuthill
Computing Services, University of California, Berkeley, CA 94720

This talk described the speaker's experiences teaching *awk* to students in the Humanities, and his perceptions of *awk* as a language. He finds *awk* to be excellent for teaching students how to program for the first time. It requires no variable declarations, has no complex I/O or file-handling routines, and includes excellent control flow facilities. One drawback of *awk* is that it is non-procedural, so students never learn how to write their own functions. Also, *awk* makes it difficult to access single array elements, and does not provide a method for opening several files at once.

Awk has particularly poor error messages and will core dump for certain types of syntax errors. Mr. Tuthill wrote a preprocessor for *awk* that checks for, and gives useful messages about, programming errors. The preprocessor can be placed in a directory that comes earlier in the users search path than where *awk* resides. If a program contains no syntax errors the front-end will *exec awk* and exit. If there are fatal errors in the program the syntax checker will exit without calling *awk*.

Awk appears to be a reasonable replacement for *snobol* for string manipulation. It appears to be simpler to learn for the beginner, and easier for the Pascal programmer to adjust to. Unfortunately, strings are limited in size, and the user has no control over these limits. Many programs fail because fields or records of input data are too long. If these limits were enlarged, or if they could be user-specified and dynamically allocated at run time, *awk* would be far more usable.

The speaker has submitted a paper for the Conference Proceedings.

NUnix Window System Description

Jack A. Test
Laboratory for Computer Science, Massachusetts Institute of Technology, Room 414, 545 Technology Square, Cambridge, Mass 02139

The NUnix Window System is a set of software that provides a user a basic window management mechanism on a high resolution display. It was developed for use with the MIT Real Time Systems Group NU Personal Computer, a 68000-based machine which uses a 820×1024 point raster-scan display, keyboard, and mouse for the user interface. The NU machines are being used for developing a multi-font editing system and drawing facility, in several circuit design projects, and in the development of new operating system concepts.

The user may create multiple overlapping rectangular windows on the display. Each is associated with an independent UNIX teletype device and a display device. A window may have up to eight independent and changeable font maps. The windows are controlled with *ioctl* calls and special signals. These allow such actions as creating a new window, drawing on it, selecting or changing the fonts associated with it, reading the state of the mouse, obtaining and/or changing the state of the window, etc. Each window belongs to a process-group to which a signal is sent whenever there is a change to the attributes of the window. There is a window-manager program which makes use of the mouse device to allow the user to select functions from a set of displayed menus. The user also has access to the display bitmap and a special graphics routine library.

The NUnix Window System is implemented in a set of device-driver routines in the UNIX V7 kernel. Most of the window driver code is machine independent with the exception of two low-level routines for driving the raster display and keyboard devices respectively. The NUnix Window System provides a basic window management mechanism that: (1) is transparent to the vast majority of user programs, (2) provides a clean user interface without the addition of any new system calls, and (3) allows user processes to manage their windows independently and with minimal kernel-imposed limitations.

The code for the NUnix Window System is available from MIT if you have a UNIX V7 license.

The speaker has submitted a paper for the Conference Proceedings.

Session Chair: Mike O'Dell, Lawrence Berkeley Laboratory

Design of an Intelligent Bitmap Terminal

Rich Fortier and *Tony Lake*
Bolt Beranek and Newman Inc., 10 Moulton St., Cambridge, Mass 02238

This talk described the design issues encountered in developing BB&N's intelligent bitmap terminal, the BitGraph. This was followed by a description of the BitGraph architecture and some discussion of the applications that have been developed.

A need was perceived for a high-resolution bitmap terminal that could be used in a variety of applications on a time-sharing system. Some aspects of the interactive user interface were to be implemented in the terminal. Compatibility with some existing classes of character and graphics terminals was consistered important so existing applications software could be used.

The various types of display devices offer trade-offs in versatility, speed, and cost. On a vector stroke terminal any point or vector can be drawn by moving the electron beam of the CRT to the desired points. Deletion of part of an image requires blanking the whole screen and redrawing the remaining image. Raster terminals continuously paint the screen, turning the electron beam on or off for each bit (like a TV). Raster character terminals display predefined characters only at predefined positions on the display. Raster bitmap terminals, on the other hand, have each point of the display represented in the terminal in RAM (random access memory)†. Thus any shape may be displayed by setting the appropriate bits in RAM. This type of terminal has become more cost effective because RAM is become much cheaper all the time.

With a raster bitmap terminal it is possible, for example, to display characters in different sizes and fonts and to integrate drawings, graphs, etc. into the text. It is also possible to to display multiple windows [as were discussed in the talk on the NUnix Window System above]. Such uses require software in the main computer and in the terminal.

A major problem in the design of a bitmap terminal is deciding how much of the functionality of the terminal is to be in hardware and how much is to be in software. The type of functions that might be useful include character generation, rectangle filling, vector generation, scrolling, buffering the video data, etc. High performance processors provide enough power to perform almost all such functions using code that is part of the terminal, rather than using hardware.

The BitGraph uses a Motorola MC68000 processor with a high speed parallel bus. The bus is connected to RAM for the display buffer, EPROM for BitGraph data and instructions, EAROM for non-

†There was an article on Smalltalk 80 in the August, 1981, issue of *Byte* Magazine.

†The types of bitmap terminals discussed at the meeting display around 1024 pixels (picture elements, or points on the display) horizontally and around 800 vertically; the BitGraph has 1024×768 pixels.

volatile storage of terminal parameters, three medium speed serial communications interfaces (for the host and keyboard, and one spare), one high speed synchronous interface which supports a variety of protocols, a programmable sound generator circuit, and an 8-bit peripheral processor and mouse interface. Additional RAM can be added to store multiple screen images.

The code in the BitGraph — the firmware — has been designed to be useful to user software that tailors the functionality of the terminal to specific applications. These routines provide a variety of functions including emulation of existing terminals. User software has full access to hardware and firmware capabilities.

The BitGraph is currently being used in UNIX-based applications in word processing, computer aided logic and VLSI design, distributed processing, and various other things. A BitGraph costs $4995 in quantity one.

The speaker has submitted a paper for the Conference Proceedings.

The SUN Workstation

Andreas Bechtolsheim
Sun Microsystems, 2310 Walsh Ave., Santa Clara, CA 95051

The SUN Workstation is a graphics-oriented personal computer running UNIX. It was designed to be used as a general purpose work station in a distributed computing environment. The user interface consists of a 800×1024 pixel bitmap display with a "RasterOP" mechanism for high-speed display updates and an optional mouse. The processor is a Motorola 68000 running on the Intel Multibus[‡]. It will be upgraded to a 68010 when it becomes available. (The 68010 will allow full virtual memory capability.) An optional local network using Ethernet is available. A group of Workstations may be networked to provide sharing of, for example, the file system and/or peripherals such printers, tapes, etc.

The Workstation currently runs Unisoft's UNIX V7, which includes some of the Berkeley enhancements. It will be upgraded to 4.2BSD when that system is available. This will allow Workstations on a network to run without a local disk, accessing files and programs from a file server on the network.

The operating system includes low-level support for graphics and window management and an implementation of the ACM CORE graphics specification plus extensions. The graphics library supports both the standard bitmap display and an optional 640×480 pixel color display.

Merging Bitmap Graphics and UNIX

Rob Pike
Bell Labs 2C-521, Murray Hill, NJ 07974

[I missed most of this talk and no paper was submitted; the abstract is reproduced below, followed by a few notes...Ed]

UNIX is a multiprogramming system. A user can run several programs simultaneously, programs that can interact (such as programs in a pipeline) or can be independent (such as parallel compilations). The syntax and semantics of pipelined processes provides a powerful and convenient form of multiprogramming, but UNIX's current mechanisms for dealing with parallel independently executing programs are weak: there is no convenient way to control several processes. The "job control" mechanism of the C-shell merely provides a way to describe which of several processes the user wishes to work with now, and does not provide a capability for letting several programs run simultaneously without interfering with, say, each other's terminal I/O.

Bitmap displays, as they have been traditionally used, take natural first step towards controlling a multiprogramming system. "Window systems" enable a user to store multiple program contexts on the same screen, but they have only (with a couple of exceptions) been static contexts, and are therefore incapable of handling multiprogramming.

[‡]Multibus is a trademark of the Intel Corporation.

The extension of the window system idea to supporting multiprogramming is simple conceptually, but surprisingly difficult to implement. There are interesting problems to solve in the areas of

- inter-process communication
- graphics support
- user interface.

This talk will address the problem and how they have been solved on the Blit terminal, a bitmap display built specifically for improving the user interface to UNIX. It will also discuss some of the extensions of the ideas developed to other areas, such as game playing and text editing.

Definition:

An intelligent terminal is not a smart-ass terminal;
it is one you can educate.

The Blit terminal is a research project, not a product. It does not have any hardware support for graphics because the 68000 can do what is needed fast enough. Mr. Pike has found that 256 Kbytes of RAM is not enough memory.

A videotape demonstrating the Blit was shown; it received a large round of applause.

Session Chair: Lauren Weinstein, Independent Consultant

The Multiple Device Queuing System

Douglas P. Kingston III and *Michael J. Muuss*
Ballistics Research Laboratory, Attn: DRDAR-BLV-V Aberdeen Proving Grounds, MD 21005

The Multiple Device Queuing System (MDQS) provides a generalized queuing mechanism for a variety of devices and (logical) queues. It has been designed with portability, expandability, robustness, and data integrity as key goals. There are provisions for resource accounting and integration into a network environment. It provides a uniform user interface where possible, while still allowing for specification of device-dependent options. Requests may be prioritized and the time of execution of the request may be specified. Queues may be displayed and modified, and requests may be deleted or restarted, even after processing has begun. MDQS replaces the functions of *lpr*(1) and *at*(1) on the normal UNIX V7 system.

MDQS is designed around a central queue which is managed by a single privileged daemon. Requests are queued by non-privileged programs and the queue control and data files are owned by the submitter. The queue control file contains queue-independent and queue-dependent information. When the requested device or job stream becomes available, the daemon executes an appropriate server process to handle the request.

The system manager can creat, modify or delete queues without the loss of requests. MDQS recognizes and supports both multiple devices per queue and multiple queues per device by mapping input for a logical device to an appropriate physical output device. MDQS also provides for crash recovery.

The MDQS system is expected to be available soon. It is designed to be compilable with a standard V7 UNIX system. Information on availability will be announced in *netnews* [and hopefully in *;login:*].

Everything you wanted to know about System III but Bell was afraid to tell you

Brian Lucas and *Mark Kampe*
Interactive Systems Corp., 1212 7th Street, Santa Monica, CA 90401

This talk made an interesting point about the groups within Bell that have been responsible for the development of recent UNIX systems:

$$\begin{tikzcd}
\text{Research} & \text{Development} \\
\text{V6} \arrow[r, dash] \arrow[d] & \text{PWB} \arrow[d] \\
\text{V7/32V} \arrow[r, dash] & \text{PWB 2.0} \arrow[d] \\
 & \text{System III} \arrow[d] \\
 & \text{System IV} \\
 & \text{etc.}
\end{tikzcd}$$

Some of the many differences between System III (S3) and V7 and PWB were discussed from a system programmers point of view, and many comments and warnings were made. [Unfortunately the slides for this talk were not available so some of the information presented is not reproduced here...Ed]

- The accounting is more production oriented.
- *Stty* and *gtty* are gone and *ioctl* is changed a lot from V7. There is some left over kernel code commented "for temporary use only" but it does not work adequately.
- The hashed process wait tables of V7 are gone.
- There is elaborate powerfail code that is suppose to work on PDPs (but not on VAXs) but it is commented out.
- Floating point hardware is strongly recommended.
- The RP06 driver still does not support multiple controllers.
- The TU77 tape drive does run with the VAX.

Bad Block Handling on the BBN C-Machine

Steve Dyer
BBN Computer Corp., 33 Moulton St., Cambridge, MA 02238

This talk discussed several methods for handling bad disk blocks on UNIX and the solution developed for BBN Computer's C machines. Historically UNIX ignored the problem of bad spots on disks: users were required to purchase expensive error-free disks and hope that they did not develop bad spots. One method developed involved removing bad blocks from the free list of the file system so they would not be allocated to any file. This was relatively easy to implement and could be used with any disk hardware, as the disk controller did not have to know about bad blocks. However, certain areas of the disk still must be free of bad blocks and non-file-system operations in general (like disk-to-disk copy) will not be prevented from using bad blocks.

Another method involves creating a dummy file for each bad sector with *mknod*. The bad sector is listed as the first data block in the inode for that file.

A more general solution to the problem of bad blocks requires an approach at a level lower than file system operations. The method called "revectoring" relies on bad sectors being marked in the header field for the sector. When an I/O request is made on such a marked sector, the request is aborted and redone on a sector in an area previously reserved for such circumstances. Marking bad sectors can be done when the disk is formatted or when an error first appears. The latter is preferable, but difficult to implement.

DEC has specified hardware and software standards for revectoring bad blocks and 4.1BSD has code implementing that standard for the RKO7, RPO6, RMO3, and RM8O disk drives.

On the C machine revectoring is implemented in the C-level disk driver and in the microcode of the disk controllers.

The speaker has submitted a paper for the Conference Proceedings. It discusses revectoring in 4.1BSD and the BBN C machine in more detail.

The New *Curses* and *Terminfo* Package

Mark Horton
Bell Laboratories, 6200 E. Broad St., Columbus, OH 43213

[Mr. Horton was presented with an award by the President of the Ad Hoc Committee on Capricious Awards (AHCCA), Mr. Greg Chesson. Unfortunately the text of the award was not available for replication in this Newsletter...Ed]

Terminfo is a database that contains descriptions of the capabilities of various terminal models and configurations. It is based on the *termcap*(5) database but contains improvements and extensions that allow description of more types of terminals and more terminal attributes. Use of many more of the features of the latest "smart" terminals will be possible with *terminfo*. *Termcap* and *terminfo* are not compatible: many of the capabilities have been renamed to correspond with ANSI standard X3.64 and parameterized strings are now supported and used (e.g., in the **cm** (cursor move) capability). *Terminfo* is a compiled database; this allows faster startup than with *termcap*.

Curses is a package of subroutines which is used to present and change an image on a full-duplex, alphanumeric, video terminal. It presents a high level model of the screen to the programmer and takes care of problems such as the type of terminal being used and optimization of output. It was designed to be relatively system-independent.

A new *curses* package has been written to implement new features and use *terminfo* and to make *curses* run with internal Bell versions of UNIX as well as 4.1BSD. Among the new features are:

- use of insert and delete line and character terminal capabilities;
- arbitrary combinations of video attributes are allowed;
- use of more than one type of terminal at a time is supported;
- improved handling of arrow and function keys;
- a programmer-accessible scrolling region; and
- a subset mini-*curses*.

The new *terminfo* and *curses* packages have not been released by Bell Laboratories.

The speaker has submitted a paper for the Conference Proceedings. It describes changes, extensions, and conversion in detail.

Programdb: Maintaining Symbol Use Data for Source Code Control

Douglas I. Kalish
Logical Software, Inc., 55 Wheeler St., Cambridge, MA 02138

Programdb is a collection of UNIX commands and database queries that provides information on the structure of a program. It uses the LOGIX relational database management system. Programdb maintains a database of pertinent information (including comments from the code) about all program symbols and modules. Reports may be generated to list the symbols in each module, which modules call which symbols, where each symbol is defined, etc. UNIX library calls and undefined or unused symbols may be easily spotted from such reports. Dependency lists may be generated to describe which modules are required by other modules for symbol definitions. These lists can be used as input to *makefile*(1) to maintain an accurate and up-to-date list of dependencies.

A UNIX shell procedure collects name list data and size information for object files, extracts identifying comments from source files and loads the data into LOGIX relations. Subsequent command procedures containing UNIX and LOGIX commands build the database from joins, projects and selects of the stored data. LOGIX queries are provided which report symbol definitions and uses either individually for a given symbol, for all symbols defined in a given module, for all symbols with a particular lexical content, or for all symbols in the database.

Programdb is available for $500 to holders of LOGIX licenses.

The Logical Softshell: A full-screen interface to UNIX

Fred M. Katz
Logical Software, Inc., 55 Wheeler St., Cambridge, MA 02138

The Logical Softshell, or "L-shell", provides a full screen interface for application programs and UNIX commands. Shell scripts and programs direct the L-shell which invoked them to display screens and to read or write "panels" within those screens.

The L-shell allows users to invoke commands and specify parameters in either of two modes: scroll mode, in which the user furnishes the familiar UNIX command lines; or full-screen mode, in which the user selects from menus, fills in forms, and toggles through options.

The application program or shell script may create, destroy or specify panels and which is active and when to read a panel. The L-shell decides which active panels to display and the layout and style of each panel. Communication with an L-shell is through pipes: messages to an L-shell are specially formatted and sent by special commands and C routines; messages from an L-shell are written to the pipe.

There are two implementations of the L-shell. In the first, the L-shell acts as a "shell's shell"; the L-shell runs as a separate process which intervenes between the terminal and some existing shell program (such as the Bourne shell or the C-shell). In the second, the L-shell assumes all of the functions of the Bourne shell and calls commands directly.

Session Chair: Mike O'Brien, Rand Corporation

MASCOT and UNIX: Their Combination and Applications

David M. Tilbrook and *Ken Jackson*
Systems Designer Ltd., Systems House, 1 Pembroke Road, Camberley, Surrey, England

MASCOT stands for Modular Approach to Software Construction Operation and Test. It provides a formalism for the expression of the software structure of a real time or parallel processing system that is independent of the hardware and language used. It also provides a methodology based on this formalism that is applicable to all parts of the life-cycle of the system. Part of MASCOT is a "kernel" which provides run-time facilities to support the formalism and scheduling and synchronization. This kernel may be implemented on a bare machine or placed on top of or integrated into an operating system.

The MASCOT formalism is represented using ACP Diagrams to display Activities, processes that can execute in parallel with all other activities, Channels, data flows between producers and a consumers, and Pools, accumulations of data that can be referenced by more than one activity. A system's ACP diagram is directly transformed into the construction commands that create that system, thus ensuring a congruence between the design and the product.

ANGUS is a a MASCOT system built into UNIX to provide a foundation for a variety of office system applications and for software engineering aids such as host/target development systems and configuration management. It was discussed in a talk on Thursday.

The speaker has submitted a paper for the Conference Proceedings. It describes the MASCOT formalism in some detail.

How to Use Lots of Memory

Michael C. Tilson
Human Computing Resources Corp., 10 St. Mary St., Toronto, Ontario, Canada M4Y 1P9

This talk described ways to use memory instead of disk on UNIX systems to reduce the amount of, and time used for, I/O. The first thing to do is to increase the amount of memory available until no swaps are being done. The second idea is to use part of memory for (part of) your *tmp* files. For example, the first *N* blocks of an RM05 could be mapped into memory. The code for this is relatively simple; the trick is that the disk drivers never look at the primary device number. The code is available in *net.sources* and from HCR. Substantial performance improvements have been noted using these techniques, at very low cost.

A High-Performance Computer System Suited to UNIX

Charles Minter
Interactive Systems, 1212 7th St., Santa Monica, CA 90401

[I missed this talk and there was no paper submitted; the abstract is reproduced below...Ed.]

This talk described a computer system designed to run UNIX. The system consists of a Motorola 68000 CPU with memory management and a cache, error correcting memory, and intelligent I/O processors.

The memory management unit uses a segmentation system optimized for speed. The grain size is 4 Kbytes and the segment size is between 4 Kbytes and 1 Mbyte. Other memory management organizations were considered but were rejected for performance reasons.

The cache size can be 4 Kbytes, 8 Kbytes, or 16 Kbytes. Both the cache and the data path to main memory is 32-bits wide. The cache uses a one-way set-associative design with write-through. Physical addresses are used in the cache address match logic to minimize cache flushes. The cache is highly integrated with the memory management unit to maximize performance. A 68000 memory read that produces a cache hit requires no CPU wait states.

One, and optionally two, data channels can support simultaneous data transfers of over 1 Mbyte per second each. These transfers are passed through the cache logic so that the cache contents remain valid after DMA transfers. These channels would be used by two disks or a disk and a high-speed network. Off-card adapters connect these channels to either an ANSI X3T9 8-inch Winchester interface or an SMD interface.

Slower speed I/O devices are connected through a special 16-bit I/O bus designed for easy interfacing. The expectation is that all devices on this bus will be controlled by a local processor that preprocesses and buffers data. The main CPU can transfer data from the I/O bus either with programmed transfers or with DMA transfers. These transfers also go through the cache logic so that, like the high-speed channel interfaces, cache data remains valid.

We have run some simple benchmarks written in C on this system. The performance varied from between 40% to 120% of a VAX 11/780. The only benchmark where the performance was not very near to or somewhat better than that of a VAX 11/780 was one that used 32-bit integer multiplies. The 68000 has no such instruction while the VAX does. It would certainly be possible to construct programs where the performance relative to the VAX would be even worse than the 40% number above. Most benchmarks we ran showed a performance of from 110% to 120% of a VAX 11/780. We have run no floating point benchmarks but since the system has no floating point hardware the performance should be substantially worse than the performance of the other benchmarks.

Perkin-Elmer's Hardware/I-O System: Flexibility That Matches UNIX

Joel R. Carter
The Wollongong Group, Inc., 1135A San Antonio Road, Palo Alto, CA 94303

This talk described Perkin-Elmer's 32-bit computers in the context of UNIX. UNIX is available for all of Perkin-Elmer's 32-bit machines through The Wollongong Group or directly from Perkin-Elmer.

Perkin-Elmer machines have two types of external communication buses: the MUX bus for medium speed devices and the EDMA bus for interfacing secondary storage devices like disk and tape.

The MUX bus can support up to 1023 devices. It has 4 interrupt levels, each with its own set of 16 32-bit registers to speed context switching between levels. There is a microcoded auto driver channel for transferring multiplexed blocks of data between a MUX bus device and memory at high speeds. There is also a bus switch which can be used to allow multiple processors to access a common bus.

The EDMA bus provides DMA to up to eight high-speed ports. There may be up to four EDMA buses on a machine. A selector channel controls each port and can support up to 16 device controllers. Maximum transfer rates are 5.7 Mbytes (not bits) when writing to memory and 8 Mbytes when reading from memory. Custom-designed interfaces are implemented through a universal DMA interface.

Multiple processors are supported through the MUX bus switch mentioned above and by allowing processors to share a common bank of memory. Programs executed from shared memory run at main memory speed.

Several applications using Perkin-Elmer 32-bit machines and specific hardware features were discussed.

Introduction to UNIX - Videotape

Fred Gerkin
Bell Laboratories Public Relations Office, 150 JFK Parkway, Short Hills, NJ 07974

[I missed this presentation...Ed]

This 22 minute videotape provides some discussion of the philosophy behind UNIX and its major features as viewed by its designers.

Thursday, July 8, 1982

Session Chair: Bob Marsh, Plexus Computers

News from USENIX

Lou Katz

The following persons were elected and assumed office at the end of the Boston Meeting:

President	Lou Katz	Directors	Bruce Borden
Vice-President	John Donnelly		Alan Nemeth
Secretary	Lew Law		Deborah K. Scherrer
Treasurer	Thomas Ferrin		Wally Wedel

A request for proposals to run the USENIX Office is being prepared and will be mailed to all persons who express interest to the USENIX Office. The deadline for receipt of proposals will be around the end of July, and the new office is expected to be open around the first of September. [The new USENIX Association Office contract has been awarded; there is an article about it elsewhere in this issue of *;login:*...Ed]

The Board would appreciate feedback from the membership concerning the joint meeting and the parallel sessions on Friday.

News from /usr/group

Bob Marsh

July was the end of the first fiscal year for /usr/group. There were 1028 members at the time of the meeting, of which about 300 were general members.

The following people were elected to the /usr/group Board of Directors on July 6, 1982:

Bob Marsh	President
Mike Florio	Executive Vice-President
Joel Carter	Directors
Judy Ross	
Roger Sippl	
Mark Ursino	

The second UNIX product catalog is being printed [and has since been mailed to the members...Ed].

The feelings about the Boston meeting that had been reported to Mr. Marsh at the meeting were:

- fewer meetings are good;
- there was a heavy weighting towards USENIX- (as opposed to /usr/group-) type talks, although the opportunities were there for /usr/group members;
- people certainly want to hold joint meetings with USENIX again.

News from AT&T

Larry Isley and *Bob Guffy*
Technology Licensing Manager, American Telephone and Telegraph, PO Box 25000, Greensboro, NC 27420

They have added five people to the licensing staff and have cleared up part, although not all, of the backlog. The licensing procedure, and time required for each step, is:

(1) written request from customer to AT&T

(2) preparation of agreement by AT&T (allow 2 to 4 weeks)

(3) agreement execution and payment (customer time)

(4) agreement execution by AT&T (1 day)

(5) software delivery (allow 1 to 5 days)

Bell has released UNIX System TSS. The source license costs $100,000 for the first CPU and $20,000 for each additional CPU.

As of July 1 the number of licensees for each type of source license was:

M-UNIX	125
UNIX-V6	499
PWB-UNIX	180
UNIX-V7	510
UNIX-32V	252
UNIX-SIII	88

This totals 1654 source licensees; they have a total of 3797 installations. [The number of binary licensees was not given...Ed]

AT&T is going to offer educational and administrative licenses for System III. The fee is going up to cover most of the administrative and maintenance costs. The **proposed** educational service fees for domestic universities are $3,000 for the initial CPU and $1,250 for each additional CPU if Bell supplies the software and $1,000 if Bell does not supply the software. Also proposed is that any domestic educational institution may upgrade all currently licensed CPU's for a one-time service fee of $12,000; this option would be available on a one-time license update only. The **proposed** administrative license fees for domestic universities are $16,000 for the initial CPU and $5,400 for each additional CPU, with full credit for upgrades of current administrative CPU's subject to a minimum of $1,000. [Educational and administrative licenses for System III are now available; the fees are described elsewhere in this issue of *;login:*.]

The question of fee-free educational use of UNIX/1100 and UNIX System TSS is under review. The question of System III documentation availability to non-licensees is being looked into. Writer's Workbench and other things are being looked at by the internal review board. [None of these had been resolved by mid-September...Ed]

They have retained a firm (Hill & Associates) to conduct a survey of educational licensees.

Presentation to Mel Ferentz

Dr. Mel Ferentz, one of the founders and the first treasurer of the USENIX Association and the first editor of *;login:*, was presented with a copy of the first edition of the UNIX manual by Dennis Ritchie. Dr. Ferentz received a long ovation from the attendees.

News from DEC

Armando Stettner
Digital Equipment Corporation, Continental Blvd., Merrimack, NH 03054

[Mr. Greg Chesson, Vice President of AHCCA, had the Flying Rubber Duck Award presented to Mr. Stettner in recognition of his reputation for "style".]

DEC has a UNIX V7 tape available free to Bell licensees that supports all standard (DEC) peripherals and all PDP-11s with memory management (11/23, /24, /34, /40, /44, /45, /55, /60, /70). Release 2.1 of the distribution, which is called **V7M**, is described as "a full source distribution of UNIX V7 plus enhancements developed by the UNIX Systems Engineering Group at Digital". Further information on the distribution, including a list of supported hardware and notes on the changes from release 1 of V7M, can be obtained from

Wendy Murphy
UNIX Distribution Coordinator
Mail Stop MK1-1 D29
Digital Equipment Corporation,
Continental Blvd.
Merrimack, NH 03054

Introduction of the EDUCOM-UNIX Task Force

Carolyn Autrey-Hunley and *Jack McCredie*

[Unfortunately I have only brief notes on this talk...Ed]

EDUCOM is an association of about 500 colleges and universities. The task force is concerned with making UNIX easier to use for general purpose computing in educational environments.

What's New at Purdue/EE Department

George Goble
Purdue University, Electrical Engineering Department, West Lafayette, IN 47907

The Purdue Electrical Engineering School has now built four dual-processor VAX 11/780s, of which three are currently running. They are currently evaluating several largish disks.

[This project was originally reported on in Volume 7, Number 1 of *;login:*. On their machine the two processors are connected and communicate through the SBI. This enables both processors to access all peripherals and memory, resulting in higher utilization of the available disk I/O and memory bandwidth. In contrast, the VAX 11/782 has multi-ported memory but otherwise the processors run as two separate machines. They run a modified version of 4.1BSD. Throughput has been around 1.9× that of a single processor 11/780.]

The speaker has submitted a paper for the Conference Proceedings. It covers the reasons for and method of converting a standard 11/780 into a dual processor system. There is a helpful discussion on the effect of user software on performance. There is also a parts list — the Spring, 1981, list cost of parts for conversion was about $66,000.

Rogue: Where It Has Been, Why It Was There, And Why It Shouldn't Have Been There In The First Place

Michael C. Toy, Esquire and *Kenneth C.R.C. Arnold*
Computer Science Division, EECS, University of California, Berkeley, CA 94720

(Rogue is a visual CRT-based fantasy game which runs under UNIX. Rogue differs from most computer fantasy games in that it is screen oriented. Commands are all a few key strokes (as opposed to pseudo-English sentences) and the results of your commands are displayed graphically on the screen rather than being explained in words.)

Rogue was written to be fun, even after you have been through it once. In one setting rogue usage had accumulated enough CPU time in three months to be third in CPU use for the whole year. Thus its usage had to be limited.

Limiting games can be done through restrictions on the time of day that they are available, lowering their priority, and through peer pressure. Mr. Arnold argued that it is reasonable to allow games on a system: they are a fact of life (people enjoy writing and/or playing them), game writing is a good educational tool, and games do provide a way of relieving frustration.

The latest version of rogue is 5.2.

UNIX etc. at National Instruments

Jeffrey L. Kodosky
National Instruments, 12109 Technology Blvd., Austin, TX 78759

The IEEE 488 bus has been called the General Purpose Instrumentation (or Interface) Bus (GPIB). National Instruments (NI) makes both DMA and non-DMA GPIB interfaces to the Unibus and Qbus. They also supply supporting software for DEC operating systems and UNIX. They perceived that the features of the GPIB that make it useful as an instrumentation bus would also make it suitable as a medium high speed interprocessor link. They have a recently announced software product dubbed NET488, which supports file transfers between any combination of processors, operating systems, and interfaces on the GPIB. Their in-house GPIB-based network connects an 11/34 running UNIX V6, an IBM Personel Computer, an 11/04 running RT, and a S100-based CP/M system. Some of the connections are made through NI-developed GPIB extenders, which allow connections to the GPIB from beyond the 20 meter length limit of the GPIB standard.

Software development for National Instruments hardware projects relies heavily on the GPIB, UNIX, and C. Each processor on their network is a development system (of sorts) for a current or special project. Software development consists of writing a minimal GPIB download program (sometimes in parallel with hardware development of the GPIB interface), procuring a C cross compiler to run on the PDP-11/34, and writing a library of routines to interface to the special hardware on the target system (e.g., fuel pump and credit card readers, waveform synthesizer and timing hardware).

The speaker has submitted a paper for the Conference Proceedings.

News From Perkin-Elmer

David J. Preston
Perkin-Elmer, Computer Systems Division, 106 Apple Street, Tinton Falls, NJ 07724

The talk presented some history of UNIX on Perkin-Elmer (P-E) computers and information on P-E's UNIX offering, Edition VII Workbench, which was developed by and is a product of the Wollongong Group, Inc. It is a V7 system with Source Code Control System, Berkeley enhancements, the RAND editor (*e*) and mail handler (*mh*), and P-E architecture-specific improvements. There are also tools to migrate software between Edition VII and P-E's OS/32. The system is supported by P-E.

There are over 50 Edition VIII Workbench installations. They typically support about 25 users in development and/or research environments.

Perkin-Elmer plans to advertise the availability of operational third party packages through a resource referral catalog.

Current Database Research at the Computer Systems Research Group, University of Toronto

John Z. Kornatowski and *Ivor Ladd*
Rhodnius Incorporated, PO Box 1, Station D, Scarborough, Ontario, Canada M1R 4Y7

The three main areas of database research are database theory and applications, office information systems theory and applications, and a major project called the Message Management Systems project that combines the research of the other two areas.

In the database theory area they are looking for ways to detect and classify cycles in the entity relationship model. In the applications area they are working mostly with the user interface. Among the projects is QBE, a full screen, semi-graphical interface that uses a subset of IBM's QBE. It allows users to specify database operations by canonical example.

Research in office information systems is in the areas of modeling an office, analysis of information and data flow, and storing and searching large message files.

The Message Management System combines research in databases, networking, offices, and electronic mail. It is concerned with both user and automatic manipulation of messages, and integration of text, voice, and image communication. The system will use UNIX, a relational database management system (Mistress), an office forms system and forms query language on top of the database, and a message management database interface and automatic message routing. The system will run on personal bitmap computers networked with a VAX.

A line-oriented interactive database editor patterned on *ed* and *awk* is being developed. There is also a C preprocessor that allows embedding queries in C programs.

Session Chair: Lou Katz, U.C. Berkeley

The UCSD MSG System: Iterative Design in the UNIX Environment

Philip J. Mercurio
Cognitive Science Lab, UCSD, C-015, La Jolla, CA 92093

This talk focused on two facets of the speaker's work on the human interface to a computer mail package: the design process and the features that were developed. The package being worked on, a modified version of RAND Corporation's MSG, was already fully functional at the beginning of the project.

An iterative design technique was used. This involved first getting suggestions from the users of the package and producing a new version of the system. Then a pool of volunteer users tested each subsequent version of the system. This iterative design technique produced a human interface preferred by many users of the original version. Some of the features developed are:

- feedback to avoid performing unwanted irreversible actions;
- customized interfaces via user option files;
- multi-level on-line "help" messages;
- increased compatibility with UNIX's "delivermail" and ARPA's "RFC 733" formats;
- establishment and maintenance of a database of mail aliases available across a number of networked machines;
- machine aliases to allow to the user to direct mail to a distant machine by giving only that machine's name, without requiring knowledge of the intervening machines.

A new user's manual is being developed using design techniques developed by Robert Brien. First a list of tasks that an expert user should be capable of performing is created. Each task is then represented in the form of a procedural "schema" that lists the current state or condition, the goal, the action, and the result. Given such a list of schemas that need to be installed in the student's mind, the next step is to perform a backwards analysis of the schemas, breaking them down into component concepts. These are assembled into superordinate schema. This process is iterated until schemas believed to be already possessed by the naive user are reached. Next, appropriate analogies which employ these are developed and assembled into a primitive mental model of how the system works. A manual which will convey this model is then written.

The speaker has submitted a paper for the Conference Proceedings.

ATLAS Test Language

Mark T. Horbal
UNIQ Computer Corp., 28 South Water St., Batavia, IL 60510

This talk described an implementation of a compiler for the ATLAS test language under UNIX. ATLAS is a language for engineers to use for writing electronic test programs. It has extensive internal checking to make it more reliable than low-level English-like specification languages.

Yacc is used to parse an ATLAS program. They had trouble getting better messages than "syntax error" and found that only 127 token were allowed by their version of *yacc*.

UNIX and Manufacturing Testing

Jack Dixon
UNIQ Computer Corp., 28 South Water St., Batavia, IL 60510

This talk described experiences implementing UNIX as the operating system on a series of computers used to test complex electronic assemblies. Each of the test computers is a PDP-11/34 with 256 Kbytes of memory, two 10 Mbyte disk drives, an asynchronous multiplexer, and various electronic test equipment interfaced through a GPIB (IEEE standard 488) bus. The test programs were written in ATLAS. Their experiences are summarized below.

- UNIX is OK for real-time applications if you understand its limitations.
- The 11/34 does not have separate instruction and data spaces and UNIX does not support overlayed load modules so processes are limited to 64 Kbytes. They implemented a "virtual memory" system at the application level to overcome this.
- They had need of a general I/O server that would handle requests from several sources. The restriction that *pipes* are limited to processes sharing common ancestors forced them to implement named pipes, which are manually created at installation time.
- The limited number of signals that can be safely used for inter-process communication is a problem.
- The lack of asynchronous non-blocking I/O requires convoluted solutions if it is needed.
- Support for esoteric RS-232 devices through *ioctl* and the various forms of the *open* call was done with surprisingly little difficulty.
- Performance can be improved significantly by good algorithms and data structures, use of register variables, being careful of which processes compete for CPU and I/O resources (including daemons and processes started by *cron*), and using *prof*(1) to detect expensive function calls (e.g., in loops).

The speaker has submitted a paper for the Conference Proceedings.

Development of a Commercial Applications System Under UNIX

Curtis Sanford and *David Walden*
BBN Computer Corp., 33 Moulton St., Cambridge, MA 02238

This talk described the use of BBN's UNIX for the development of a commercial applications package and some of the tools and techniques developed. The package developed was *InfoMail* (**Infor**mation Management and Electronic **Mail**), a portable message handling and filing system. It runs on BBN C-Machines, VAX/VMS, and IBM VM/CMS and MVS/TSO. It provides a menu interface with full screen interaction as well as a command interface and on-line "help" facilities.

Portability was obtained by using a superset of *ratfor* called *rat* that can generate code for either *ratfor* or for PL/1. *Rat* maintains a database of information on subprograms to enforce proper calling and returns and provides parameterized macros and detection of non-portable constructs.

They have developed a source control system that helps maintain a software change log, provides for development of parts in isolation from the rest of the parts, allows incremental integration, and permits system "snapshots".

They found UNIX to be highly cost effective for development of non-UNIX-targeted commercial applications.

Application Programming Environment on UNIX

Masatoshi Kurihara and *Yukio Ikadai*
Software Research Associates, Inc., 9314 Cherry Hill Road #519, College Park, MD 20740

Software Research Associates, Inc. (SRA) is a large, 15 year old independent software house in Japan. Roughly 70% of SRA's work is development of business data processing application programs using COBOL, and the rest is development of process control application programs in Fortran or assembly languages.

Two years ago the company decided to change the programming style from the traditional paper-and-pencil approach to an on-line interactive environment. They installed a VAX running UNIX with 3BSD. Since then, various experiments have been made to construct a practical environment for average application programmers. Some of those are listed below.

- A "Module Testbed for Fortran" (verification suite) was developed. It was used by a pilot project to increase the programmers' productivity and to improve the quality of resulting programs; the results indicated more than a 2× improvement in productivity over traditional methods.
- A "Strictly-Controlled Environment for Novice Programmers" was developed to provide a group of easy-to-use commands for novice programmers who had no knowledge about UNIX.
- A "COBOL Interpreter and Testing Tools" environment was developed [on UNIX!].

The installation of UNIX has increased productivity and product reliability in many cases. However, the they have also found that the average programmer has some difficulty using UNIX and that the change to an interactive environment has created needs for new management styles.

Managing A Roomful of UNIX Systems

Benjamin J. Woznick
BBN Computer Corp., 33 Moulton St., Cambridge, MA 02238

BBN's Computer Services Division's UNIX Computer Center runs about 18 UNIX systems. Some of the machines are connected to ARPANET, others to a BBN network, and still others to both networks. This talk described some of the services provided and problems encountered by the UNIX Computer Center.

The fundamental service provided is the operation of the hardware. One of the objectives is to provide as uniform a service as possible across all the machines. This required developing procedures for almost everything.

- Administrative requests are handled by electronic mail to a single mailbox. The reader forwards these requests to appropriate persons, relieving them of having to deal directly with the users.
- Most bug reports are similarly sent to a single mailbox for resolution or distribution by the reader. There are a few additional mailboxes for important subsystems. They have found that acknowledging receipt of a bug is more important than scheduling an immediate fix of it.
- Software distribution to the various machines has been a problem. Providing each system with a tape drive seemed impractical. Disk or network transfer of *tar* files has not been fully satisfactory. A new system is being developed that does a controlled match of directories over the network.
- System reboot has been helped by the addition of the Berkeley robustness patches, super-block rewrite, and an automatic file-system checker and repair program.
- Backups are done to punched cards@Backups are a continuing problem. Disk-to-disk backups use a lot of the file space but are operationally simple. Network transfers for backup require either a lot of time or a very large bandwidth, particularly for full system backups; then there is the question of where to put the data. It appears to them that an automated incremental backup over a very fast network to 6250 bpi tapes is probably the best solution currently available.
- Originally they tried living without a systems programmer; they have recently hired one.

Given the current state of the technology they feel that multiple mini-computers are a reasonable alternative to a giant main-frame, which would not allow groups to control their own environments, or a plethora of personal computers, with their implicit barrier to the sharing of files and programs.

The speaker has submitted a paper for the Conference Proceedings.

Session Chair: Marleen Martin, 3COM Corporation

On Ring Architected Local Networks

Howard Salwen
Proteon Associates, Inc., 24 Crescent St., Waltham, MA 02154

This talk described ring architectured local networks in general, and Proteon's implementation of such a network for MIT. The ring is implemented as a passive wire center with attachments made by relays (CTL's) which are powered by their host. Each relay connects to a host-specific board (HSB) that provides a DMA interface to the host. To join the network a host energizes its relay; otherwise the relay remains passive in the ring. Each active CTL acts as a repeater of the data on the ring unless it is originating or receiving a message.

When the ring is idle a "token" control character is circulated. When a message is to be sent the CTL modifies the token to signal the beginning of a message. Messages are sent to specific 8-bit node addresses. Their length is variable, up to a maximum of 2044 bytes, with an end of message indicator terminating the data. A token is sent at the end of the message to allow other nodes to place messages on the ring. When a message is received it is modified by the receivers CTL. This is seen by the sender when the message comes back around the ring; it then removes the message.

The speaker has submitted a paper for the Conference Proceedings. It describes in more detail the functioning of a ring network.

A Family of Portable Systems Based on System III

Heinz Lycklama and *Steve Zucker*
Interactive Systems, 1212 7th St., Santa Monica, CA 90401

Interactive is developing a line of products based on UNIX System III. The products, generically called IS/3, are targeted to run on a broad range of processors including the Zilog Z8000 and Motorola 68000 as well as the DEC VAX-11s and PDP-11s. Goals for the systems include:

- Full System III compatibility at the programmer, as well as the user, interface.
- Ease of portability of applications not only across systems running IS/3 but to Version 7 and other selected variants of UNIX as well.
- Enhanced performance, robustness, and generality.
- Improved system management tools.

In order to avoid the "least common denominator" syndrome, which makes it possible to take advantage of features that are not available on all the systems, they have defined a programming environment and a set of programming standards for applications that masks the differences between the systems with porting libraries and *include* files.

The major performance enhancement to date has been the parameterization of disk block size to permit the throughput improvements achieved by Berkeley on systems where the space lost to fragmentation is tolerable, the memory for larger buffers is available, and the hardware can support larger transfers. For the VAX, additional device and adapter support has also been added based on the work at Berkeley.

The system management tools have been improved not so much by changing what they do but how they are told what to do. For example, all the file system utilities — *fsck, df, mount, mkfs, dump, restor,* etc. — make use of an attribute-value file (*/etc/filesystems*) which specify all defaults on a per-file system basis. This makes it much easier to change file system layout, lessens the burden on the system manager to remember details such as device names and disk interleaving factors and the like, and consolidates the information that is normally scattered about in separate files for each utility, in various manual entries, and in the system manager's head. Another attribute-value file (*/etc/ports*) consolidates the information associated with ports and provides control over port modes and usage.

Current Status of Mistress (Version 2) and Future Plans

John Z. Kornatowski and *Ivor Ladd*
Rhodnius Inc., PO Box 1, Station D, Scarborough, Ontario, Canada M1R 4Y7

Mistress is a relational database management system. Among its features: it allows multiple concurrent users, uses a SQL-like query language, allows interactive data entry and update, provides for bulk loading and unloading of data, support for many data types, on-line *help*, and a built-in command and data editor. Mistress runs under V6 and later UNIX systems and on Xenix, Onyx, IS/*, and Zeus.

The latest version of Mistress is 2.1, released in July. It is fully upward compatible with version 1. Highlights include: built-in sorting capability; creating relations on the fly (results of queries placed in new relations); a simple macro facility to save tedious typing; range checking on data entry; and a new text data type (variable-length record). There is also a new report generator. A new set of Host-Language Interface routines has been written to simplify the low-level programming language interface.

Mistress has been ported to several new machines, including a Eastern Electric Wringer Washer, Serial# AJ203774K, running Cistern 3 BRANDX. The washer version is available only under a special un-educational profit-only license. [The distribution should be requested on tape instead of disk as the adjustment of the wringer heads is less critical...Ed]

C Compiler for Data General AOS/VS

Robert Weisman and *Mike Meissner*
Data General Corp., 4400 Computer Dr., Westboro, MA 01580

This talk described Data General's AOS/VS C compiler for their 32-bit Eclipse MV/family computers. The compiler is a complete implementation of C as specified by Kernighan and Ritchie, with documented extensions. It uses AOS/VS common compiler components consisting of the code generator, optimizer, source level debugger, cross reference facility, and runtime environment. The common runtime environment allows programs written in C to call, and be called by, programs written in other AOS/VS languages.

Most UNIX V7 system calls are emulated by a library layered on top of AOS/VS. A package of buffered I/O routines that are fairly close to the "standard I/O" package are layered on top of the UNIX system calls.

Several extensions have been made to the compiler. Some library functions generate in-line code. An optional cross reference facility is part of the compiler so the user can see the references in the context of the compilation. There are also human-oriented error messages.

Debugging is aided by source and assembly language level debuggers.

Code compiled by the C compiler can also run under AOS/RT32, a subset of AOS/VS oriented to real-time applications.

The speaker has submitted a paper for the Conference Proceedings.

Systems Designers Limited Vendor Presentation on Angus

Elwyn Wareham
Systems Designers Ltd., Systems House, 1 Pembroke Road, Camberley, Surrey, England

[I missed this talk and there was no paper submitted; the abstract is reproduced below...Ed.]

The presentation will outline a new product from Systems Designers Limited, which extends the facilities available under UNIX by adding new tools aimed at the development of real-time, parallel processing systems.

The background of the company, particularly in the development of such tools, will be outlined with particular emphasis on involvement in Mascot, (Modular Approach to Software Construction, Operation and Test[†]) the method which underpins the tools being offered, and which is widely used for real-time systems development in Europe.

Example applications, and the associated benefits of using the new tools will be outlined. Extensions to the initial product range and the additional capabilities offered will be discussed.

UTS: UNIX on the Amdahl 470

Daniel Walsh
Amdahl Corporation, PO Box 470, Sunnyvale, CA 94086

The talk discussed UTS, Amdahl's UNIX V7-based product, and their experiences porting V7. UTS runs in a virtual machine under VM/370 on Amdahl and Amdahl-compatible computers.

UTS was originally developed for internal use for applications development. They started with an early version of the Bell C/370 compiler; they now use an improved version of the V7 *pcc*. UTS does not support full duplex terminals so they created a 3270 screen editor based on *ed*.

They have made several improvements related to efficiency under VM and to provide UTS-to-other system communications. Other changes include an I/O configurator, an automatic disk checker and fixer, a simple *sccs* replacement, an *nroff* front end, an interactive spelling checker, and a C pre-processor for 3270 applications. Their in-house operation supports about 450 users on a 470/V8; they routinely have 160 users logged in at a time. They find that VM crashes more often than UTS.

Consideration is being given to System III support, full duplex ASCII support, Ethernet support, and native mode operation (they figure there is 20-30% overhead in using VM).

UTS can be licensed from Amdahl for $1,000 per month for universities or $1,500 per month for commercial installations. This price includes support by Amdahl. To run UTS you will need a 4341-2 equivalent computer, VM/370, and a UNIX V7 license.

[†]Mascot was described in a talk on Wednesday.

Friday, July 9, 1982

Editor's Note

The notes for Friday's talks were prepared by

Bill Tuthill
Computing Services, University of California, Berkeley, CA 94720

Session Chair: Clem Cole, U.C. Berkeley

Compact Data Analysis Programs for UNIX

Gary Perlman
Dept. of Psychology, C009, University of California at San Diego, La Jolla, CA 92093

Most data analysis packages have programs that can do more than the analysis desired, including data validation, data transformations, and specific analyses. A problem with this approach is that individual programs with all such capabilities built in are so large as to be unusable on mini-computers without significant space and time (for overlaying) penalties. In the spirit of the UNIX philosophy of combining special purpose modules via pipelines, Mr. Perlman designed a small set of data analysis programs. Instead of incorporating elaborate data validation and transforming procedures into every program, a few small programs do the job. A typical data analysis session repeats the following abstract shell script:

```
validate < data
transform < data | analysis
```

After validation and transformations, data can be analyzed by an analysis of variance, multiple regression, or descriptive statistics for univariate or bivariate distributions. Because of some serious human engineering, all these programs have elegant user interfaces that for the most part eliminate the requirement of specifying design information. These programs have acquired acceptance over the past two years of use at about 40 UNIX installations.

MENUNIX: An Interface to UNIX Programs and Files

Gary Perlman
Dept. of Psychology, C009, University of California at San Diego La Jolla, CA 92093

MENUNIX is a menu driven interface to UNIX, a possible substitute for the shell. Different windows on the screen are devoted to user files, system programs, command history, and status. Furthermore, UNIX commands are divided into workbenches of related utilities.

The FILE MENU presents a window into the files of the current directory and displays the file names, sizes, and a special coding of the access protection that indicates file ownership. Files and directories are distinguished by highlighting as well as the protection code. The PROGRAM MENU displays an extensible content hierarchy of the programs of UNIX with clusters of related programs (called workbenches) for tasks such as manipulating the file system, programming (with sub-workbenches for C, FORTRAN, Pascal, and LISP programming), writing papers, sending and receiving electronic mail, and so on. Because the PROGRAM MENU hierarchy is extensible, new programs and workbenches can be incorporated with ease (for example, they have a workbench for doing statistics). One of the biggest advantages of MENUNIX is being able to search through workbenches to find commands of interest, a task not well supported by the UNIX manual. Workbenches contain all and only those commands related to a task, and these workbenches are a cognitively manageable size, making it possible to become expert in limited areas of UNIX.

MENUNIX is an experimental interface in which the psychological problems of user-interface design have been studied. Both psychological theory and data collected under controlled experimental conditions were applied to its design. Its problems are poor screen handling and perhaps too much complexity.

Description of a Menu Creation and Interpretation System

Michael J. Heffler
Delft Consulting Corporation, 392 Bleecker St., New York, NY 10014

The Menu Creation and Interpretation System (MCIS) is a menu system development tool that can provide the high-level control structure and the user interface for any application program or software system. MCIS simplifies the task of building menu-driven systems while presenting a user interface that is easily learned and used. It can be used with any program (for example, *mail*), not just as a substitute shell.

Human factors considerations have been stressed in the system design. A user can move from menu to menu, obtain help when needed, and easily recover from errors. Menu systems created with MCIS are extendable and easily modified, enabling them to meet changing user needs. In addition, through the specification of user profiles, menu systems can be customized to meet the needs of individual users.

MCIS is written in the C programming language under the UNIX operating system. It can be used in any environment that includes a C compiler and the UNIX standard I/O library. Since it scrolls the screen, it can be used on any type of terminal.

Benchmarking to Eliminate the Benchwarmers

Eugene F. Dronek
Aim Technology, 3333 Bowers Avenue, Suite 199, Santa Clara, CA 95051

In response to the dearth of UNIX benchmarking tools, a portable suite of benchmark programs has been developed to measure performance of different hardware configurations, and of different UNIX systems on the market. Several approaches to benchmarking were rejected, and Mr. Dronek finally settled on an approach that attempted to measure specific UNIX requirements.

These benchmark programs measure and display system throughput in specific areas — disk, memory, interprocess communication, floating point, C compilation, command completeness, and multiuser response. There are plotting routines included, some of which graph user response (stretch) under simulated multi-user loads. Examples were given of benchmarks done on VAX 750, PDP 11/70 and M68000 machines.

This package is available for $500 from Aim Technology.

A UNIX Benchmarking Tool, and Results from the PDP-11/44, VAX 11/780, and Perkin-Elmer 3242

Martin Tuori
D.C.I.E.M., P.O. Box 2000, Downsview, Ontario, Canada, M3M 389

A UNIX benchmarking tool has been developed, which can run variable numbers of four different tasks: an editor, *nroff*, the C compiler, and floating point processes. The implementation is based on command files, and is easily modified or extended. It can be used to exercise a system, to provide timing information, and to check the consistency of the results obtained from each process.

This benchmarking tool has been used to compare several UNIX systems, including a PDP-11/44, and VAX 11/780, and a Perkin-Elmer 3242. The two 32 bit systems are configured almost identically, having 3MB main memory, three 300MB discs on two controllers, and floating point processor. The VAX runs 4BSD, while the P-E runs TWG's UNIX Edition VII. The current results suggest that the P-E is slightly faster at the user component, while the VAX's speed at the system component brings its real time to about 10% less than on the P-E, although the P-E did better on floating point tests. Multiuser degradation was not bad on either machine. However, these results may be attributable to operating system differences.

This tool will be made available to all UNIX users through USENIX.

UTS: UNIX on the Amdahl 470

Daniel Walsh
Amdahl Corporation, P.O. Box 470, Sunnyvale, CA 94086

This talk discussed the porting of UNIX to run on the Amdahl 470 computer. Major points included: the history of the port, going back to 1975; design decisions in transporting UNIX to this architecture; efficiency improvements; and the successes and failures of the port.

Amdahl UNIX now has a replacement for SCCS, the NED screen editor with C and spell interfaces, a facility for making makefiles, and a program to draw charts of C programs. Recent improvements include demand paging, fixes to f77, Pascal, and improvements to C. The future might bring System III support, native mode, and full duplex communications.

At Amdahl there are 500 in-house users of UTS, with 150-170 simultaneous users. The Amdahl 470 compiles the 21,000 lines of kernel code in 4 minutes. UTS is quite reliable; in fact, it crashes less frequently than VM.

Porting UNIX to a Personal Computer

Gregory J. O'Brien
Digital Equipment Corporation, ZK1-3/B21, 110 Spit Brook Rd., Nashua, NH 03062

This talk described the results of a project to port UNIX V7 to DEC's PC-350. With 256K bytes of memory and a 22-bit address space, this personal computer performs much like low-end PDP processors. It comes with a reconfigurable bitmap display, with shading or color. However, no 9-track tape or casette is available; backup is done on floppy disk.

The operating system for a dedicated personal computer must meet different requirements than the OS for a timesharing computer. Some features of the kernel (multi-user support) were removed for the sake of compactness. PC-350 uses the same instruction set processor as a PDP 11/23, so object code generated on the 11/23 could be run without recompilation. The system has the same ISP but a new bus, a new interrupt controller, and all new peripherals. Finally, the kernel uses up around 47K of memory, and performance is about 70% as good as an 11/23.

UNIX/Prime: Porting the UNIX operating system to Prime machines

James L. Weiner and *Brian L. Johnson*
Computer Science Department, University of New Hampshire, Durham, NH 03824

A project is under way at the Computer Science Department of the University of New Hampshire to port UNIX to run on Prime Equipment. The idea is that UNIX should run on top of the kernel of the Prime operating system (PRIMOS[†]), rather than on a bare bones machine or on top of the full operating system. UNIX will have the same access to kernel operations that PRIMOS has and thus its implementation will not suffer from having to be simulated. This arrangement allows UNIX to take advantage of aids offered by the PRIMOS kernel. Shared memory routines and new I/O routines will have to be implemented. The goal is to develop a standard UNIX without restrictions. Of course, due to the differing hardware there will be some exceptions.

The Prime hardware makes the project interesting. Memory on the Prime is partitioned into 65K segments, with three rings of protection. Every user runs a virtual machine with security provided by the rings. Prime also has hardware that supports many of the same features as does the UNIX notion of signals. They are actually more flexible since users may define their own. Along with the hardware support for signals is hardware support for semaphores, which this implementation uses.

[*]PRIMOS is a trademark of Prime Computer.

Part of the project involves developing a general mechanism for treating segments of memory as an abstract data type that supports operations such as allocate, deallocate and share. This will allow an additional form of interprocess communication with no additional overhead, and will also be a start towards incorporating datagrams into a UNIX network.

Currently, everything is completed and partially tested, except that process control is missing. The filesystem is now about as fast as the VAX filesystem. It was very helpful that the Prime has an imbedded operating system; this made implementation easier.

Session Chair: Richard Bratt, BBN Computer Corp.

UNIX Emulation, Again

Sanand Patel and *Richard Sniderman*
Human Computing Resources Corp., 10 St. Mary St., Toronto, Ontario, Canada M4Y 1P9

VX is a software package that emulates the UNIX environment under the VAX/VMS operating system. The UNIX shell is invoked from that system's standard command interpreter, providing access to all UNIX utilities. Essentially all of the UNIX system calls are provided, enabling most UNIX C programs to run without modification. No changes to the host system are necessary.

VX is a substantial improvement over similar packages, such as Eunice, in the degree to which UNIX is emulated. In particular, *fork*, *exec*, inherited file descriptors, and full UNIX file names are all handled as in a true UNIX environment. VX is also more reliable then Eunice.

Many difficult implementation problems had to be solved, in particular process management, file management, and file name translation.

Process creation was tricky because VMS spawns a new image, rather than forking two copies of the current process. Pipes were difficult to implement because VMS does not allow sharing of files between processes, and has no open file table. VMS filenames are 9 characters long with a 3 character suffix, and case is not significant; maximum directory depth is 8.

A UNIX Emulator for VAX/VMS

Michael Caplinger
Department of Math Sciences, Rice University, P.O. Box 1892, Houston, TX 77001

Software performance and portability issues, (in particular, the poor performance of the UNIX Fortran compiler) have forced many VAX sites to reluctantly choose VMS over UNIX. However, the programming environment provided by VMS is much less convenient and user-oriented than the UNIX environment, as well as being unfamiliar and restrictive to someone familiar with UNIX.

One solution to the problem (the "Virtual Operation System" (VOS) of Lawrence Berkeley Labs) provides a UNIX-like interface on top of an already existing operating system. It does not, however, address the problem of porting UNIX programs. Using a different approach, David Kashtan of SRI wrote "Eunice", a UNIX simulator. He attempted to solve the porting problem by emulating UNIX at the system call level. Eunice has several problems. Some of the system calls are implemented inefficiently (and some incorrectly). Also, many factors make large amounts of source-level modification necessary to run arbitrary UNIX programs.

Faced with this problem, Rice University is in the process of designing and implementing a complete emulation of UNIX, called Phoenix, consisting of a set of system calls and a new loader developed for VMS. The system calls provide exact simulations of the facilities provided by the UNIX kernel, even providing functions that are not provided by VMS. The loader takes UNIX object modules and produces a VMS executable. This gives Phoenix complete object-level compatibility with UNIX.

The development of Phoenix is being done in a somewhat-enhanced version of Eunice (which already uses the new loader). The completion of this project should make porting of nearly all UNIX programs to VMS a trivial exercise, and will extend the benefits of the UNIX community to VMS sites who, up to now, have been denied access to them. The software is in the public domain, and a tape is available for $200.

Selecting a DBMS for a Super Micro

Mike Bender
ZILOG, Building B1-3, 1315 Dell Avenue, Campbell, CA 95008

Zilog considered various approaches to Database Management on the System 8000 (a 16-bit UNIX machine). Among the DBMSs studied were Oracle, Ingres, Informix, Mistress, Unify, Logix, Sequitur, MDBS III, and Data Base Plus.

A Super Micro must support a large range of database sizes and have full networking capabilities. Some users may want to have a Data Base Machine residing on a network, others may want to distribute their data, and finally there will be those that simply want a stand-alone database. It is important that a DBMS support these possibilities.

To fulfill these capabilities a DBMS has to be more than just efficient, easy to use, etc. There are a few unique features that it should have:

- It must support a relational model, so that the data can be logically divided between machines.
- It must be divisible into a disjoint front-end and back-end so that the user can sit at a different machine from the data.
- It must be dictionary-driven, to help the user with the addressing, and so that integrity control can be centralized.
- And it should be able to work as a stand-alone system, independent of the network.

The speaker claimed that Zilog had chosen a particular database system, but he did not specify which one.

A Business-Oriented File Manager under UNIX, with Contention Control and ISAM

Gary Williams
Durango Systems, Inc., 3003 North First St., San Jose, CA 95134

Many standard applications programs require record-oriented sequential and random access files, and indexed sequential access (ISAM) files, with extensive record-contention control not provided by the standard UNIX file management system. The addition of a pseudo device driver, /dev/locker, and the use of a set of file management routines written in C provide file and record level access control without modifying the standard UNIX file system or penalizing nonapplications users.

Using the contention-control system involves a status request, possibly a sleep, a lock, and an unlock. There is support for variable-record sequential access files, for fixed-record random or sequential access files, and for ISAM files using a B-tree index with pleat and fold routines to maintain balance.

IAFORM, An On-Screen Definition Package for Data Retrieval Forms

T. Scott Pyne
Computer Systems Technology Division, Science Applications, Inc., 1710 Goodridge Drive, P.O. Box 1303, McLean, VA 22101

The usefulness of DBMSs has been enhanced by on-screen query processors, which allow a user to fill in a form, initiate queries, and view retrieved records. ORACLE, a relational DBMS from RSI, provides such a facility in its IAP (Interactive Applications Processor), but the forms used must be defined via a tedious question-and-answer process. It's necessary to answer 60-100 questions, and often takes two or three tries to get it right. SAI has developed a package, called IAFORM, which allows a user to define a form with the on-screen query system by drawing a picture directly on the screen using a subset of a standard text editor (the Rand editor).

The editor takes input in a manner such as "32c" (meaning an alphabetic field of length 32) and draws a field on the screen consisting of the appropriate number of lower case characters (in the example case, 32 'c' characters would be shown). The editor allows field definitions to be picked up and moved around, so the elements of a form may be juxtaposed after initial definition. Once the user has defined the form to his/her satisfaction, the template is translated into a set of description structures from which a question and answer file is built. This file is submitted to ORACLE's forms definition utility and produces an input file for use with the on-screen query processor.

At present, only a few locations are running the UNIX version of ORACLE.

Tabstar - Information Data Base Management

Gordon W. Waidhofer
The Wollongong Group, Inc., 1135A San Antonio Road, Palo Alto, CA 94303

Tabstar is a collection of utilities for simple database manipulation and report generation. Tabstar addresses the range of problems that are technically infeasible with standard UNIX utilities (e.g. grep, uniq, sed) alone, and are economically infeasible with full scale database systems (e.g. oracle, ingres).

A database is a text file, manipulated by the Tabstar and existing UNIX utilities. Database views are created as shell command files, and may be used in the same way as database files. The design strategy of Tabstar is to take as great advantage of existing UNIX text manipulation and test processing programs as possible. There are additional facilities for table searching and sorting. Some of the existing applications are marketing prospect tracking, action item tracking, and documentation control.

Session Chair: Doug Michels, The Santa Cruz Operation

Is UNIX as a Standard Doomed?

Robert B. Greenberg
4318 Collins Ct., #3, Mountain View, CA 94040

[Mr. Greenburg did not show up in time for this talk, so his abstract is printed verbatim.]

A tremendous ground swell of interest in UNIX by the commercial community began late last decade. The continued growth of excitement is due to UNIX's two principal assets: it has a technically superior design for software development environments, and it is a very likely choice for becoming the operating system standard for low-end business systems.

Last summer, this author discussed many of the shortcomings of UNIX from a technical standpoint. In this talk, he pursues the second point: just how realistic is it that UNIX will be THE operating system standard. The talk concerns itself with the lack of guidance in the UNIX community, and the over-simplicity of the desired goals. The relative immaturity exhibited in such a new marketplace, and the current and anticipated practices and events that will ultimately determine the outcome of the current UNIX rage.

The talk is intended to dispel the implicit assumption that UNIX will automatically become the standard (or that it needs to be), criticize some current marketing claims, and remind the newly initiated as to some of UNIX's weaknesses. The focus is to discuss what needs to be done to insure that UNIX has a healthy future in the non-computer environment. The author has a great love for UNIX, but feels this love is best served by prophesying warnings against over-optimistic views of UNIX as the panacea of computing for the masses.

A Survey of UNIX Usage in Scientific and Business Applications

James R. Hanley and *Jeffry A. Scott*
Laboratory for Information and Science in Agriculture, Room 302 Aylesworth Hall, Colorado State University, Ft. Collins, CO 80523

[Mr. Hanley and Mr. Scott did not show up at all, so their abstract is printed verbatim.]

Increasing attention is being focused on UNIX to determine its appropriateness as a general purpose operating system for scientific and business applications. Unfortunately, little of this information has made its way into the literature. As a result, a survey was conducted among UNIX users and followers to determine:

(1) To what degree UNIX is currently being used for scientific and business applications within the United States and at what level of expertise.

(2) How the UNIX user community views the strengths, weakness, and competitive position of UNIX as a general purpose operating system for scientific and business applications.

The survey's sample was drawn from the ranks of the /usr/group and USENIX memberships. This paper presents the results of the survey and the conclusions drawn.

The Coming UNIX Crash

Roger McKee
Vice President, The Wollongong Group, Inc., 1135A San Antonio Road, Palo Alto, CA 94303

[Mr. McKee participated in a panel discussion, which will be described below. He did not give this talk, but the abstract is printed verbatim anyway.]

The history of data processing abounds with examples of companies and their offerings that were valuable and useful. Some of these companies did not survive; their offerings did not become industry standards. The reasons for failure vary. This paper examines the reasons and applies them to the world of today's UNIX companies. We present several obstacles that have to be overcome before UNIX can become a standard in the data processing community.

The Commercialization of UNIX

Rebecca Thomas and *Jean Yates*
Yates Ventures, Inc.

[Due to illness, neither Ms. Thomas nor Ms. Yates were present, and there was no paper submitted. Their abstract is printed verbatim.]

UNIX is evolving rapidly from a research-oriented product with limited commercial appeal to a product with overwhelming impact on the data processing industry. In our presentation, we will show the implications of this at two levels:

(1) Technical direction of UNIX — what kinds of features will be developed — will they leave universities out in the cold?

(2) Market impact — what will UNIX's place be in the office automation / microcomputer / minicomputer markets.

At the technical level, Ms. Thomas will look at the enhancements and improvements that the USENIX community wants, and compare them to what commercial vendors are doing. She will look at the possible directions that Bell Labs will take in view of a more commercial orientation for UNIX inside Bell.

At the market level, Ms. Yates will present results of Gnostic Concepts' intensive market analysis of existing and new markets for UNIX. This data is part of an $18,000 study and has never been presented to the public before.

Panel Discussion

Roger McKee of Wollongong pointed out that a DEC or Perkin-Elmer operating system would cost $35,000 with only a third of what UNIX has in the way of languages and utilities. He also maintained that UNIX is what's in the manual — everything there should be on any vendor's system.

Bob Greenberg enumerated some of the reasons why UNIX as it stands lacks commercial viability. First, the software it offers is not what's needed in the marketplace. Second, some necessary software it does offer could be easily duplicated elsewhere. Third, although the operating system itself is portable, existing applications are not portable to UNIX because there's no Cobol, no real Basic, and the Fortran isn't good. Fourth, UNIX is not upwardly-compatible with the current standard, CP/M. Finally, the pricing structure is inconsistent (selling more machines has a non-linear relation with decreased fees), and the owner is not the marketer.

Roger Sippel of Relational Database Systems maintained that UNIX standards are not really necessary. He thinks UNIX will be the base for quality applications, but that the user won't see it behind the applications in 1986. The idea of CP/M is more important than CP/M itself — an unfriendly, incomplete system — because it provides a software base.

The panel discussion gave way to questions from the audience. People from Bell Labs and Universities challenged the vendors to provide adequate software support, and to sell complete systems with all the utilities. People from the commercial world lamented that Berkeley was already changing the kernel out from under them, in effect creating a moving target of standardization. The new filesystem with 256 character filenames came in for the most criticism.

Conference Proceedings

A conference proceedings that will contain copies of the papers submitted at the Boston meeting is being prepared by /usr/group. It will be available this fall at cost. An announcement will be printed in *;login:* and elsewhere when the proceedings are available.

Assistance Needed

The following requests for assistance were received at the Association Office. If you can help with any of these problems please contact the person listed **and** send the information to the Association Office.

C compiler for IBM 4341 running VM/SP
Apparently C/370 runs only under the MVS operating system
N. Hauser
Yale University
A. W. Wright Nuclear Structure Lab
P.O. Box 6666
272 Whitney Ave.
New Haven, Conn 06511

ONYX C8002 communications with IBM, NCR, and Texas Computers
Any usable solutions would be welcome
G. G. Lange
Abcos Computers (Pty) Ltd.
P.O. BOX 7369
Johannesburg, 2000
South Africa

IBM 3270 emulator for use on UNIX
Mitchell Schoenbrun or John Mullen
Phaser Software Systems
Suite 406
126 Post Street
San Francisco, CA 94108
(415) 421-1627

Basic language processors for V7/32V
Galen Jarvinen
University of California
Office of Finance and Planning
Santa Cruz, CA 95064

The Skeptic

- Un-attributable sources are planting the rumor that DEC has finally decided to get serious about UNIX. Apparently they are trying to solidify their position in the UNIX market by offering more than just hardware and the semi-official offerings of TIG. The sources indicate that DEC will supply field service and software support for their V7M2 distribution by the end of '82 and for 4.2BSD sometime next year. This is suppose to include binary licenses. The sources also indicate that DEC will implement VNX-like products across its full line from personal computers through its 16-, 32-, and 36-bit machines.
- There are published rumors of new UNIX projects at Bell: specifically a version that is more user-friendly and more suited to microcomputer business applications. Reportedly these would be available for licensing sometime in 1982. Is Bell doing a port to the 68000 too? Or are they going to try to sell the 3B? Officials at Bell have declined to comment.

[This month's Skeptic writes with Wordstar on an Osborne from a fishing camp in Missouri, the "show me" state. Contributions for this column are welcome; send them to the the Editor at the Association Office.]

USENIX Association
Application for New Individual or Public Membership for 1982

[ ] Individual — Non-disclosure Covered by Institution with Source License: $12

Institutional Affiliation: ______________________________

Nature of Affiliation: ______________________________

[ ] Individual — Non-disclosure Covered by Institution with Binary License: $12

Institutional Affiliation: ______________________________

Nature of Affiliation: ______________________________

[ ] Public — Not covered by Non-Disclosure: $12

Mailing Address (Individual Members must use institution address):

Name: ______________________________

____

____

____

____

Phone: ____________________

Network address: ____________________

[ ] Overseas airmail, add $5.00
[ ] Invoice required, add $3.00 bookkeeping charge
[ ] Receipt required, add $3.00 bookkeeping charge

Check enclosed: $__________

Return completed form to:

USENIX Association
P.O. Box 7
El Cerrito, CA 94530

For Institutional membership or membership renewal contact the Association at the address above.

USENIX Association
P.O. Box 7
El Cerrito, CA 94530

First Class Mail